Design and evaluation of a perceptually adaptive rendering system for immersive virtual reality environments

Kimberly Ann Weaver
Iowa State University

Follow this and additional works at: http://lib.dr.iastate.edu/rtd
Part of the Cognitive Psychology Commons, and the Computer Sciences Commons

Recommended Citation
http://lib.dr.iastate.edu/rtd/14895

This Thesis is brought to you for free and open access by Iowa State University Digital Repository. It has been accepted for inclusion in Retrospective Theses and Dissertations by an authorized administrator of Iowa State University Digital Repository. For more information, please contact digirep@iastate.edu.
Design and evaluation of a perceptually adaptive rendering system for immersive virtual reality environments

by

Kimberly Ann Weaver

A thesis submitted to the graduate faculty
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

Major: Human Computer Interaction

Program of Study Committee:
Derrick Parkhurst (Major Professor)
Chris Harding
Shana Smith

Iowa State University
Ames, Iowa
2007

Copyright © Kimberly Ann Weaver, 2007. All rights reserved.
# TABLE OF CONTENTS

LIST OF FIGURES ........................................................................................................... v
LIST OF TABLES .............................................................................................................. vii
ACKNOWLEDGEMENTS ................................................................................................. viii
ABSTRACT ......................................................................................................................... ix

CHAPTER 1. INTRODUCTION ............................................................................................. 1
1.1. Computer Graphics Background .............................................................................. 3
   1.1.1. Representing Models as Triangular Meshes ....................................................... 3
   1.1.2. The Graphics Pipeline ...................................................................................... 4
   1.1.3. Targets for Simplification ................................................................................. 7
   1.1.4. Representing Geometric Levels of Detail .......................................................... 17
1.2. Visual Perception Background ................................................................................ 20
   1.2.1. Anatomy of the Retina .................................................................................... 20
   1.2.2. Attention .......................................................................................................... 22
   1.2.3. Scene Perception ............................................................................................. 23
1.3. Level of Detail Selection Techniques ...................................................................... 26
   1.3.1. Reactive Level of Detail Selection .................................................................... 26
   1.3.2. Predictive Level of Detail Selection .................................................................. 34
   1.3.3. Summary .......................................................................................................... 35
1.4. Immersive Virtual Reality Environments ................................................................ 36
1.5. Research Approach .................................................................................................. 38

CHAPTER 2. THE RENDERING SYSTEM ........................................................................... 41
2.1. Interaction Modes ................................................................................................... 41
   2.1.1. Simulation Mode ............................................................................................. 42
   2.1.2. Exploration ..................................................................................................... 43
   2.1.3. Edit .................................................................................................................. 44
   2.1.4. Experiment ..................................................................................................... 47
2.2. Level of Detail Selection .......................................................................................... 47
   2.2.1. Eccentricity ...................................................................................................... 47
   2.2.2. Attentional Hysteresis .................................................................................... 48
2.3. Using Models .......................................................................................................... 50
   2.3.1. Model Acquisition ......................................................................................... 50
   2.3.2. Model Simplification ....................................................................................... 50
2.4. XML Scripting ......................................................................................................... 51
   2.4.1. Master Configuration ...................................................................................... 51
   2.4.2. Scene Graph Configuration .............................................................................. 52
   2.4.3. Architecture Configuration ............................................................................. 53
   2.4.4. Light Configuration ......................................................................................... 54
   2.4.5. Camera Configuration ...................................................................................... 57
APPENDIX A. SHADE LICENSE ................................................................. 131
APPENDIX B. MODELS USED IN THE PILOT AND FULL STUDIES .......... 132
APPENDIX C. VISUAL ACUITY QUESTIONNAIRE .................................. 135
APPENDIX D. SIMULATOR SICKNESS QUESTIONNAIRE ............................ 139
APPENDIX E. POST-TASK QUESTIONNAIRE ............................................ 140
APPENDIX F. END OF STUDY QUESTIONNAIRE ...................................... 141
BIBLIOGRAPHY ......................................................................................... 143
LIST OF FIGURES

Figure 1. A Triangle Mesh ................................................................. 4
Figure 2. The Graphics Pipeline ......................................................... 5
Figure 3. Comparison of Process with Serial Stages versus Parallel Stages .... 6
Figure 4. Stanford Bunny rendered as a wireframe at three levels of detail ....... 10
Figure 5. Vertex shader code for a wood texture ................................ 11
Figure 6. Fragment shader code for a wood texture ................................ 12
Figure 7. Shader simplification ......................................................... 14
Figure 8. Simulation Simplification .................................................... 16
Figure 9. Continuous Level of Detail Representation ................................ 19
Figure 10. View-Dependent LOD Representation .................................. 20
Figure 11. The Anatomy of the Eye .................................................... 21
Figure 12. Differing Scan Paths by Task .............................................. 25
Figure 13. Top-Down Schematic Representation of Distance-Based LOD Selection ...... 27
Figure 14. Implementation of Distance-Based LOD Selection .................. 28
Figure 15. Top-Down Schematic Representation of Eccentricity-Based LOD Selection ...... 29
Figure 16. Implementation of Eccentricity-Based LOD Selection ............... 31
Figure 17. Living Room Environment Used by Parkhurst and Niebur .......... 39
Figure 18. The 4 Interaction Modes .................................................... 42
Figure 19. Screenshot of the AEST Application ..................................... 43
Figure 20. Available Exploration Mode Interactions ................................ 44
Figure 21. Available Interactions for Edit Mode ..................................... 46
Figure 22. Available Interactions in Experiment Mode ............................ 47
Figure 23. Top-Down Schematic Representation of Eccentricity-Based Level of Detail ... 48
Figure 24. Structure of the XML Master Configuration File ..................... 52
Figure 25. Structure of the XML Scene Graph Configuration File ............. 55
Figure 26. Structure of the XML Architecture File ................................ 56
Figure 27. Structure of the XML Light Object File ................................ 56
Figure 28. Structure of the XML Camera Configuration File ................... 57
Figure 29. Structure of the XML Sound Configuration File ..................... 58
Figure 30. Structure of the XML Skeleton Configuration File .................. 59
Figure 31. Structure of the XML Script File ........................................ 61
Figure 32. Structure of the XML LOD Schema File .............................. 61
Figure 33. Hierarchy of Control for the Component Managers .................. 63
Figure 34. Commands Accepted by the Components in the Rendering System .... 64
Figure 35. Using Scene Time ........................................................... 65
Figure 36. Scene Graph Diagram ....................................................... 67
Figure 37. Rendering of the Original C6 ............................................. 72
Figure 38. Virtual Environment Displayed in the Closed Configuration C4 ....... 72
Figure 39. Models Used as Targets in the Pilot Study at Each of Three Levels of Detail ...... 75
Figure 40. Visualization of Object Layout in the Virtual Environment ............ 79
Figure 41. Top-Down Representation of Eccentricity ............................. 80
Figure 42. Visual Acuity Scores for Participants in the Pilot Study ................ 84
Figure 43. Comparison of Simulator Sickness Averages ........................... 86
Figure 44. Overall Simulator Sickness in the Pilot Study .........................................................87
Figure 45. Simulator Sickness Ratings in the Pilot Study Separated by Type ..................88
Figure 46. Frame Rates, Search Times and Distraction Ratings for the Pilot Study ........91
Figure 47. Responses to the End-of-the-Study Questionnaire .....................................93
Figure 48. Pilot Study Measures Separated by Eccentricity Value ............................96
Figure 49. Pilot Study Measures Separated by Hysteresis Value ..............................97
Figure 50. Full Study Dog and Dragon Target Stimuli .................................................104
Figure 51. Full Study Monkey and Snail Target Stimuli ...........................................105
Figure 52. Virtual Environment and Objects Used in the Full Study ........................107
Figure 53. Visual Acuity Responses from the Full Study ...........................................110
Figure 54. Comparison of Average Simulator Sickness Scores in Part 1 .................112
Figure 55. Comparison of Average Simulator Sickness Scores in Part 2 .....................112
Figure 56. Search Time, Frame Rates, and Distraction Ratings in Part 1 of the Full Study .115
Figure 57. Search Distance and Number of Search Errors in Part 1 of the Full Study ....116
Figure 58. Search Times, Frame Rates, and Distraction Ratings in Part 2 of the Full Study119
Figure 59. Search Distances and Number of Search Errors in Part 2 of the Full Study ....120
LIST OF TABLES

Table 1. Common LOD Representations ................................................................. 17
Table 2. Summary of Relevant LOD Research ....................................................... 36
Table 3. Experimental Conditions for Participants in the Pilot Study ..................... 82
Table 4. Example Qualitative Responses to the Conditions in the Pilot Study ............ 89
Table 5. Changes to the Experimental Protocol of the Full Study ............................ 101
ACKNOWLEDGEMENTS

Thank you to the students, staff, and faculty connected with VRAC and the HCI program for providing advice, support, and friendship over the course of my studies here. Special thanks to my lab mates in the Human and Computer Vision Lab for feedback on posters and presentations and more especially for all of the fun we had.

I would like to express my thanks to my major professor, Dr. Derrick Parkhurst. His ideas, advice and support have been invaluable over the course of my graduate studies and in particular during the writing of this thesis. Thank you for accepting me as your student, for looking out for me even when I wasn’t looking out for myself, and for giving me so many opportunities to broaden my experience.

Thank you also to my thesis committee members Dr. Chris Harding and Dr. Shana Smith, as well as to Dr. James Oliver for your valuable feedback and contributions to my research and this thesis.

I also appreciate the tremendous support from my family. Countless thanks to my parents who were always available to patiently listen to me ramble. Thank you to my brother for all of his pragmatic advice on life and grad school. Also, thank you to my sister-in-law for helping me figure out how to do my statistics even when she had many other things she needed to do.
This thesis presents the design and evaluation of a perceptually adaptive rendering system for immersive virtual reality. Rendering realistic computer generated scenes can be computationally intensive. Perceptually adaptive rendering reduces the computational burden by rendering detail only where it is needed. A rendering system was designed to employ perceptually adaptive rendering techniques in environments running in immersive virtual reality. The rendering system combines lessons learned from psychology and computer science. Eccentricity from the user’s point of gaze is used to determine when to render detail in an immersive virtual environment, and when it can be omitted. A pilot study and a full study were carried out to evaluate the efficacy of the perceptually adaptive rendering system. The studies showed that frame rates can be improved without overly distracting the user when an eccentricity-based perceptually adaptive rendering technique is employed. Perceptually adaptive rendering techniques can be applied in older systems and enable them to display higher quality environments without reducing interactivity.
CHAPTER 1. INTRODUCTION

One goal of virtual reality research is to create virtual environments that are indistinguishable from real environments. We strive for this realism because it can make people feel more immersed in the environment. They are able to interact with the virtual environment as if they are interacting with the real world. Creating these near perfect reproductions of the real world requires processing significant amounts of data and performing complex calculations very quickly. Objects from nature such as trees are very difficult to render realistically because there are so many branches and leaves, that the computer cannot process all of the data fast enough for display. The most realistic lighting cannot be used in the environment because it takes substantial computational power to determine the paths of the individual light rays to precisely calculate shadows and reflections. It is also difficult to calculate with complete accuracy the effect that complex objects have on each other when colliding because it is necessary to check for collisions against every feature of an object.

If the environment is too complex, the computer will have to spend a long time preprocessing and rendering it. As a result, the display will be updated very slowly. The delay in display rendering increases the lag between when a user initiates an interaction and when the environment updates to reflect that interaction. If this lag time is too long, the user will not have a sense of immersion. Any benefit gained from displaying a realistic environment will be neutralized. Therefore, it is necessary to find a balance between creating a realistic environment and maintaining acceptable interaction.
The balancing act becomes even more pronounced in fully immersive environments for which large portions of the scene must be rendered simultaneously, or in environments that are shared over a network where bandwidth becomes a limiting factor. Graphics cards which perform most of the calculations involved in rendering a realistic virtual environment are becoming faster. They are capable of handling even larger amounts of data concurrently. However, this hardware is still insufficient and will be for the foreseeable future. Thus, computational resources must be managed to ensure the highest quality scene with the least amount of overhead.

The goal of this research is to manage computational resources by employing knowledge gained from the study of computer graphics and visual perception to reduce environment complexity gracefully, without noticeable effects on task performance or perceived scene quality. This can be done using perceptually adaptive rendering to determine when to render detail and when to leave detail out based on our understanding of the limits of human perception. Immersive virtual reality environments require multiple views of the same scene to be rendered simultaneously, even though the user is unable to see much of the environment. Even the portion of the scene which is currently visible to the user can be simplified given that the user only pays attention to a small portion of the scene at any given time. This can greatly improve rendering update times. Although there are many different aspects of a virtual environment which can be simplified, this research employs geometric simplification. Geometric simplification has been around the longest, and therefore tools are readily available for its implementation in virtual reality environments. The reduction in geometric complexity decreases the amount of computation and processing performed in each frame so that the display can be updated quicker. The improvement in frame rate leads to improved interaction between the user and the environment.
1.1. **Computer Graphics Background**

Perceptually adaptive rendering research relies on two aspects of computer graphics. The first is the method by which models are defined in a three dimensional (3D) graphics application. The second is the nature of the graphics pipeline. Section 1.1 describes basic information about these two topics and their relation to perceptually adaptive rendering.

1.1.1. **Representing Models as Triangular Meshes**

Objects in 3D graphical environments are typically represented as a collection of vertices and edges that define a surface. Most operations are performed on the vertices. The graphics processing unit (GPU) of a computer is designed to operate most efficiently when those vertices are grouped into triangles. These triangles cannot just be arbitrarily provided to the GPU. In very large models, there could be many triangles sharing a single vertex. With no organization of triangles, that single vertex would have to be processed many separate times, leading to inefficiency. Instead, the most common way to represent a 3D model is to use a triangle mesh. When triangles are arranged into a mesh, a single vertex needs to be processed only once by the graphics card independent of the number of constituent triangles. Figure 1 shows a triangle mesh of the Stanford Bunny. The Stanford Bunny is one of the most used test models in computer graphics (Turk, The Stanford Bunny, 2000).
1.1.2. The Graphics Pipeline

The graphics or rendering pipeline is the set of processes by which a scene composed of 3D models is transformed into a two-dimensional (2D) image for display.

Figure 2 illustrates the graphics pipeline. The stages are modeling, vertex manipulation, rasterization, pixel manipulation, and display. First, the modeling stage involves the representation of an object in 3D space. It also involves associating the object with any lighting and materials. This is where properties based in simulation interactions are calculated. For example, if physics is enabled, then the positions of objects as a result of collisions need to be determined. The first stage of the graphics pipeline is not implemented in the graphics hardware, but instead runs on the central processing unit (CPU).
The remaining stages are processes that occur in the GPU. In the vertex manipulation stage, the operations that must be performed on each vertex are executed. This can include lighting and shading, as well as projecting 3D vertices onto a 2D plane. The rasterization stage is where the 2D vertex information is converted into pixels for display. This involves taking all of the vertices, transforming them into 2D points, and then filling in the 2D triangles based on these transformations. During the pixel manipulation stage, the color for each pixel in the final display is determined. This may be determined by vertex colors and textures which can be defined along with a model, or more recently by shaders. Shaders are a set of instructions which the programmer can define to override the normal GPU calculations. They can be used to calculate more accurate lighting or to add other effects that can be better represented in 2D space instead of 3D space. After all of the information for a given pixel is gathered, that pixel is sent to the display device.

Each vertex and the resulting pixels can function as independent entities. The calculations for any single vertex do not normally depend on the result of operations on neighboring vertices. The calculations for any single pixel do not normally depend on those of its neighbors. This means that once a vertex has left a stage in the pipeline, and has moved onto another stage, another vertex can be pushed into the pipeline. If calculations for one vertex depended on the results from another vertex, then the computer would have to completely process a single vertex and render it to the display before it could begin processing the next vertex. This is like washing, drying, and folding a single load of laundry before starting on a second load of laundry. If you had to do three loads of laundry without

![Figure 2. The Graphics Pipeline](image-url)
any parallel stages, then it would take nine time units to complete all of the loads of laundry. Because it is possible to start washing the second load of laundry when the first load goes into dryer, you can speed up the laundry process. When the first load is done drying, you can put the second load into the dryer, start the third load in the washer, and fold the first load. Instead of the nine time units required for processing laundry in series, it is possible to processes all three loads of laundry in five steps when stages are performed in parallel. Figure 3 illustrates the difference. The diagram on the left shows the laundry process with serial stages. The diagram on the right shows the faster laundry process with parallel stages.

![Diagram of laundry process with serial and parallel stages](image)

Figure 3. Comparison of Process with Serial Stages versus Parallel Stages
This does not mean that every portion of the graphics pipeline is actively computing at all times. Bottlenecks can occur when the processing in one stage takes longer than the processing in another stage. We see this when doing laundry too. Washing a load of laundry may only take 25 minutes, but it takes 60 minutes to dry that same load of clothing. This means that the second load of laundry will have to sit an extra 35 minutes in the washer while the dryer finishes the first load of laundry.

One bottleneck in the graphics pipeline occurs during physics calculation. Although dedicated hardware is now on the market, most systems utilize the CPU to simulate the physical environment. This is a bottleneck in the modeling stage of the graphics pipeline because it is computationally expensive to calculate the interactions between two complex meshes. Bottlenecks may also occur at the vertex manipulation stage with models containing large numbers of vertices because each vertex must be transformed into screen space and many vertices will correspond to a single pixel on the screen. There can also be bottlenecks in the pixel manipulation stage when many complex calculations are performed on each pixel, for example when the shader needs to run through calculations on a pixel multiple times to obtain the correct end color.

The GPU contains many components which can operate in parallel allowing for multiple simultaneous vertex and pixel manipulations, much in the same way that having more washing machines at a laundromat allows you to wash multiple loads of laundry at the same time. The number of vertex and pixel processing units varies greatly across graphics cards. For example, the GeForce 7950 GX2 has 16 vertex processing units and 48 pixel processing units (NVIDIA Corporation, 2007).

1.1.3. Targets for Simplification

Current graphics research focuses on using simplification to alleviate bottlenecks in three different locations in the rendering pipeline. These methods are geometric
simplification, shader simplification, and physics simplification. Depending on the nature of the environment to be rendered, these methods can be used individually, or they can be combined to minimize delays in data flow within the pipeline. Geometric and shader simplification are two methods aimed at reducing the computational load on the rendering pipeline of the GPU. Physics simplification helps simplify the calculations that must be performed on the CPU for each frame before data can be sent to the GPU for rendering.

The method of simplification which has been in use the longest is geometric simplification. Geometric simplification focuses on alleviating the bottleneck in the vertex portion of the pipeline. Because many operations must be performed on each vertex, the time it takes to render a single frame is greatly dependent upon the number of vertices that must be rendered. One of the simplest methods of reducing the number of vertices is to determine early on in the rendering process which vertices in the object will be visible at any one point in time from a single viewpoint. Vertices that will not be seen, because they define geometry on the opposite side of the model from what is currently visible, can be removed from the rendering pipeline in order to speed up the rendering process. This is called backface culling. Also, any portion of an object that is occluded by another object can also be safely removed from the rendering pipeline without any visible change in the final render. The utility of these simplifications is clear because the rendered scenes in the culled and non-culled scenarios will be identical to the viewer, and the rendering time will be reduced due to the decreased scene complexity.

In environments with millions or billions of vertices, back face and occlusion culling will not be sufficient to maintain interactivity given current hardware constraints. There are still too many vertices for the rendering pipeline to process quickly. This is where perceptually driven simplification can be useful. The simplified geometry can be obtained in many ways, and for a complete explanation for choosing a method for creating the reduced geometry see Luebke’s tutorial on algorithms for simplifying polygons (Luebke D. P., 2001).
Unfortunately, there is no one method that works for all applications. The appropriate algorithm for simplification must be selected based upon the resources that are available and the aspects of the models that are most important. Figure 4 shows the Stanford Bunny as a triangle mesh at three different levels of geometric detail that were created using the QSlim algorithm.

QSlim iteratively collapses neighboring vertex pairs to single vertices (Garland & Heckbert, 1997). Two vertices are considered pairs if they share an edge, or if they are within a minimum distance from each other. The surface error is one measure of the amount of change from the original model caused by vertex pair reduction. Errors for each vertex are computed and stored in a 4x4 matrix. The error matrix contains the sum of the two matrices that would be used to transform the original positions of the two vertices to the location of the new vertex. The algorithm attempts to find a new vertex which minimizes the error. First, the algorithm attempts to find the new vertex by solving the quadratic error equation. If the quadratic equation matrix cannot be inverted, then the algorithm tries to find the new vertex along a line between the two original vertices. If this cannot be found either, the new vertex is selected as one of the original two vertices or the midpoint between the two original vertices. The error resulting from simplification is the sum of the errors of the two original vertices.
Figure 4. Stanford Bunny rendered as a wireframe at three levels of detail.
The second type of simplification targets programmable shaders. Programmable shaders are a relatively recent development, which allow the direct programming of the GPU. Using shaders, a programmer can directly manipulate the properties of each vertex and each pixel of a model. Vertex shaders define which operations are performed in the vertex manipulation stage of the graphics pipeline and apply effects per vertex. It is not possible to add new vertices using a vertex shader, but existing vertices can be transformed and manipulated. Vertex shaders are also used to calculate attributes which can be interpolated by the fragment shaders if they are used. Fragment shaders replace the pixel manipulation stage of the graphics pipeline and apply effects per pixel. Per pixel calculations can be created to apply naturalistic-appearing materials such as wood to an object. Because calculations must be made for each pixel, extremely complex pixel shaders can slow down the rendering process even though they are executed on specialized hardware. Figure 5 and Figure 6 contain the vertex and fragment shaders used to create a wood texture on an object.

```plaintext
// Simple vertex shader for wood
// Author: John Kessenich
// Copyright (c) 2002-2004 3Dlabs Inc. Ltd.
// See Appendix A for license information

varying float lightIntensity;
varying vec3 Position;
uniform float Scale;

void main(void)
{
    vec4 pos = gl_ModelViewMatrix * gl_Vertex;
    Position = vec3(gl_Vertex) * Scale;
    vec3 LightPosition0 = gl_LightSource[0].position.xyz;
    vec3 LightPosition1 = gl_LightSource[1].position.xyz;
    vec3 tnorm = normalize(gl_NormalMatrix * gl_Normal);
    lightIntensity = max(dot(normalize(LightPosition0 - vec3(pos)),
                           tnorm), 0.0) * 0.9 + max(dot(normalize(LightPosition1 - vec3(pos)),
                           tnorm), 0.0) * 0.9;
    gl_Position = gl_ModelViewProjectionMatrix * gl_Vertex;
}
```

Figure 5. Vertex shader code for a wood texture
// Simple fragment shader for wood
// Author: John Kessenich
// Copyright (c) 2002-2004 3Dlabs Inc. Ltd.
// See Appendix A for license information

uniform float GrainSizeRecip;
uniform vec3 DarkColor;
uniform vec3 spread;

varying float lightIntensity;
varying vec3 Position;

void main (void)
{
  // cheap noise
  vec3 location = Position;
  vec3 floorvec = vec3(floor(10.0 * Position.x), 0.0, floor(10.0 * Position.z));
  vec3 noise = Position * 10.0 - floorvec - 0.5;
  noise *= noise;
  location += noise * 0.12;

  // distance from axis
  float dist = location.x * location.x + location.z * location.z;
  float grain = dist * GrainSizeRecip;

  // grain effects as function of distance
  float brightness = fract(grain);
  if (brightness > 0.5)
    brightness = (1.0 - brightness);
  vec3 color = DarkColor + brightness * spread;
  brightness = fract(grain * 7.0);
  if (brightness > 0.5)
    brightness = 1.0 - brightness;
  color -= brightness * spread;

  // also as a function of lines parallel to the axis
  brightness = fract(grain * 47.0) * 0.60;
  float line = fract(Position.z + Position.x);
  float snap = floor(line * 20.0) * (1.0/20.0);
  if (line < snap + 0.006)
    color -= brightness * spread;

  // apply lighting effects from vertex processor
  color = clamp(color * lightIntensity, 0.0, 1.0);

  gl_FragColor = vec4(color, 1.0);
}

Figure 6. Fragment shader code for a wood texture
Even when you reduce the number of vertices being sent to the GPU, the simplified object will still take up the same amount of space on the display as the original object, so the same number of pixels will need to be filled by a fragment shader. Though the overall render time will be lower due to the reduced number of vertices, the time needed to determine the color of a single pixel can be significant when the scene is lit with a complex lighting scheme or when materials are very complex. Shader simplification involves the creation of multiple shaders each with a different level of complexity. Shader simplification is an improvement to the pixel manipulation portion of the graphics pipeline. Figure 7 shows the wood shader defined in Figure 5 and Figure 6 applied to the Stanford Bunny at two levels of detail.

A significant problem associated with shader simplification is that it is a non-trivial task to create multiple shaders that are simplified from an original in a logical and continuous manner. Pellacini suggested a method of automatic shader generation in which the lines of code in a shader are programmatically simplified based on a series of rules (Pellacini, 2005). In order to create a smooth transition in areas where there would be a switch between two shader levels of detail, it would be possible to blend two neighboring shaders together. Care needs to be taken however, because it can be computationally expensive to switch between shaders.
Figure 7. Shader simplification
The third target of simplification targets physics calculations. The complex physics calculations required to create realistic interactions among irregularly shaped models can also be a cause for reduced interactivity. Simulation simplification is not a method for alleviating a bottleneck on the graphics card, but instead reduces the complexity of calculations performed by the CPU before the data is transferred to the graphics card. It is a simplification of the modeling portion of the rendering pipeline.

The physical representations of objects can be simplified to reduce the complexity of physics calculations (O'Sullivan C., Dingliana, Giang, & Kaiser, 2003). The highest level of detail representation is the mesh itself. At the next lowest level of detail, the geometric volume is filled using very small boxes where the boxes almost entirely occupy the same amount of space as the geometric representation. As the level of detail decreases, the sizes of the boxes used to fill the volume increase until at the very lowest level of detail, a single box is used. The simulation representation for the interacting objects can also be approximated using other primitive geometric objects whose motion is simple to calculate.

Another method of reducing the computation for physical simulations involves the use of an interruptible algorithm (O'Sullivan, Radach, & Collins, 1999). The application is allotted a certain amount of time in which to perform all of the collision detection calculations in each frame. Once the time cut-off is reached, the objects in the scene are placed in the positions and the orientations which had been computed. If the application had not yet completed a calculation for a specific object, then the transform would not be as accurate as those for which the calculations had been completed. This allows for improved interactivity, but the unrealistic behavior of some colliding objects is perceptible.
a) High Level of Detail Physical Representation

b) Low Level of Detail Physical Representation

Figure 8. Simulation Simplification
Because the geometric representation and the simulation representations are separate and distinct, one does not need to change both the geometric representation and the simulation representation at the same time. Instead, they can be optimized individually. Figure 8 shows two different ways a single object can be represented for physical simulation. In the top image, the bunny is represented accurately by a mesh. In the bottom image, the bunny is represented by a cube that is a simple bounding box. If all objects are simulated as boxes, interaction between objects in a scene would be much easier to calculate. In both images, the red lines indicate the representation for physical simulation. A medium level of detail physical representation could be a further simplified mesh, or it could be a series of cubes arranged to approximate the volume of the original model.

1.1.4. Representing Geometric Levels of Detail

There are three types of geometric level of detail (LOD) representations: discrete level of detail, continuous level of detail, and view-dependent level of detail (Luebke, Reddy, Cohen, Varshney, Watson, & Huebner, 2003). Choosing the appropriate representation depends on many factors including the complexity of the original unsimplified environment, the amount of memory that you have to store the representations, and the amount screen space a single object will occupy. Table 1 describes the three different types of representations.

<table>
<thead>
<tr>
<th>Representation</th>
<th>Stored As…</th>
<th>Rendered as…</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Discrete LOD</strong></td>
<td>Multiple files each of which corresponds to a single LOD</td>
<td>A group of polygons based on a single most appropriate LOD.</td>
</tr>
<tr>
<td><strong>Continuous LOD</strong></td>
<td>Single file loaded as a hierarchy of separations and joins</td>
<td>A group of polygons based on a single most appropriate LOD.</td>
</tr>
<tr>
<td><strong>View Dependent LOD</strong></td>
<td>Single file with separations and joins for specific areas of the model</td>
<td>The most appropriate LOD for each area of the object</td>
</tr>
</tbody>
</table>
Discrete level of detail representation is a method first used in flight simulators and is used in most modern video games (Luebke, Reddy, Cohen, Varshney, Watson, & Huebner, 2003). In those systems, the level of detail decreases the farther away an object is located from the observer. In the discrete level of detail representation, multiple versions of a single model are stored. Figure 4 (page 10) shows the Stanford bunny at three discrete levels of detail. Each version of the model corresponds to a level of detail which has been computed and saved beforehand. Each version of the model must be loaded into memory at run time. Because multiple versions of the same model must be kept in memory and the amount of memory available is limited, the number of different versions needs to be restricted. Otherwise, meshes for different levels of detail would have to be switched in and out of main memory. Due to the limited number of levels which can be defined, “popping” can become a big issue. Popping is a visual artifact that occurs during the switch between two different levels of detail when the differences between two levels of detail are visually noticeable (Luebke, Reddy, Cohen, Varshney, Watson, & Huebner, 2003). Popping is undesirable because it is distracting and reduces the realistic feel of the environment.

A more complex level of detail representation is continuous level of detail. In a continuous level of detail representation, an object is stored as a hierarchy of vertex splits and merges. A schematic that demonstrates this concept is shown in Figure 9. In the schematic, each circle in the tree represents a single vertex. At the bottom of the tree is the high-detail representation with 4 vertices. In the low detail representation at the top of the tree, those 4 vertices have been joined to create a single vertex. Simplification is done by combining two connected vertices into one vertex either through the elimination of one of the vertices or through creating a new vertex which is an interpolation between the two vertices and removing those two original vertices. It is only necessary to traverse the hierarchy until the appropriate level of detail is reached. The geometric complexity of a model can be changed gradually. Objects are less likely to “pop” and become more noticeable during the transition
between levels of detail because the difference in appearance between the two levels of detail is small.

![Continuous Level of Detail Representation](image)

Sometimes there will be an object that is highly detailed and is located in the middle of the field of view, taking up a large portion of the screen. With only continuous or discrete level of detail representations, the entire object would need to be rendered at the highest level of detail given that the user is looking at some part of this object. This would reduce the degree of geometric simplification that would otherwise be possible. In this situation, view-dependent level of detail representation is useful. In view-dependent level of detail representation, a single large object which spans many levels of detail in the environment can be rendered with a different level of detail for each portion of the display. When using discrete or continuous level of detail representations, an environment can have multiple levels of detail existing within the same frame, when using a view-dependent representation a single model can have multiple levels of detail existing within the same frame. Figure 10
illustrates a terrain model with view-dependent level of detail representation. It was created using ROAM (Duchaineau, 2003). The terrain in the image is stored as a single model, but by using a view-dependent level of detail representation, the area in the lower right-hand corner can be presented with more detail than the region in the upper left-hand corner.

1.2. **Visual Perception Background**

Perceptually adaptive rendering research is also based on aspects of visual perception. The first aspect is the physical anatomy of the eye and more specifically the retina. A second aspect is how people go about collecting information when perceiving a scene. The third is the role of attention in visual perception. Section 1.2 describes basic information about these three aspects and their relation to perceptually adaptive rendering.

1.2.1. **Anatomy of the Retina**
The retina, lining the back of the eye, converts light into electrical signals in the nervous system. A diagram of the eye is shown in Figure 11. Photoreceptors in the retina are responsible for transducing the light received by the eye into electrical energy. There are two types of photoreceptors: rods and cones. Rods are more numerous than cones and are able to function under very low light levels. They are broadly sensitive across the light spectrum. There are about 100 to 120 million rods on a single retina. Cones, on the other hand, are less sensitive to light, but are responsible for our ability to sense color. Cones are 1/15th as abundant as rods, numbering between 7 and 8 million in a single retina. The greatest concentration of cones is in a small area near the center of the retina called the fovea. The fovea is only 1.5 mm in diameter (Ferwerda, 2001). If you hold your thumb at arm's length, the area of the real world that is projected onto the fovea in high resolution is approximately the size of your thumbnail (Baudisch, DeCarlo, Duchowski, & Geisler, 2003). The area of the visual field that projects to the fovea covers only 2 degrees of visual angle (Cater, Chalmers, & Dalton, 2003). As a result of this non-uniform photoreceptor arrangement, very little of the visual field is actually sensed at a high resolution.

**Figure 11. The Anatomy of the Eye**
(NIH National Eye Institute, 2006)
Our ability to perceive detail decreases in relation to eccentricity, the angular distance, from the fovea. As eccentricity increases, cone density decreases nearly exponentially. By 20 degrees eccentricity, the cone density asymptotes to a minimum. Rod density is higher in the periphery than cone density. It decreases in the far periphery and approaches its minimum density at 75 to 80 degrees (Ferwerda, 2001). This is why we perceive only the coarsest information about objects located on the edges of our field of view.

1.2.2. Attention

Certain aspects of scenes are more likely than others to attract a person’s attention. Saliency is the measure of the likelihood for an object to attract one’s attention. Properties of a feature that make it more salient include brighter colors, movement, task importance, location in the foreground, inclusion of patterns or faces, and familiarity to the user (Luebke, Reddy, Cohen, Varshney, Watson, & Huebner, 2003). Although these characteristics are known to attract attention, it is clear from the differing scan paths in Figure 12 that the saliency value of a specific portion of a scene cannot be calculated merely from the inherent properties of the scene or object alone.

Although it is possible to move attention without moving our eyes, it requires more effort to move attention with fixed eyes (Rayner, 1998). Therefore, if an object captures our attention, we are highly likely to move our eyes so that the object is viewed in detail using the foveal region of the retina.

Change blindness and inattentitional blindness are two behavioral phenomena related to visual perception. Both are related to our inability to detect very significant changes in our visual surroundings. Change blindness is the “inability of the human eye to detect what should be obvious changes” (Cater, Chalmers, & Dalton, 2003). This phenomenon can occur when the visual field is briefly disrupted. A participant is presented with a scene. The scene is briefly disrupted, and when it becomes fully visible again, some aspect of the scene has
been changed. The change is often undetectable because the participant cannot compare all regions in the original and changed images (Simons, Nevarez, & Boot, 2005).

The body of change blindness research indicates that our internal representation of the external world is much less detailed than our experience of “seeing” suggests (Cater, Chalmers, & Dalton, 2003). We feel that we are able to see the entire world in complete detail, but we are really only perceiving small portions of our environment and filling in the holes with what our experience tells us should be there. Many studies of change blindness use some sort of occlusion technique: either fully covering the image, or “splashing” blocks of pixels over various parts of the image (Intille, 2002). It is also possible to mask changes by making the transition very slowly (O'Regan, Rensink, & Clark, 1999). Eye blinks are easily detectable and saccades can be determined using eye tracking, which make them good targets for exploitation. Changes to objects in the environment could be timed to occur only during a blink or a saccade. Changes to the most salient objects will tend to be detected faster (Intille, 2002).

1.2.3. Scene Perception

In everyday life, we do not notice the decrease in visual acuity in the peripheral visual field. This is because of the way we move our eyes when viewing a scene. We are constantly making quick eye movements called saccades in order to get new information about our environment. Our eyes are relatively still between saccades for about 200-300 milliseconds. That period of relative stillness is known as a fixation (Rayner, 1998). Our eyes make saccades about three times per second (Henderson, 2003).

The path the eyes take while viewing a scene is not the same for everyone. It can depend upon the task. Figure 12 shows the results of an experiment which tracked eye movements while participants viewed Repin’s ‘An Unexpected Visitor’ (Yarbus, 1967). Each picture with red lines indicates a scan path for a different task. For clarity, the scan
paths have been superimposed on the original image. The scan paths may have been aligned approximately.

At the top of Figure 12 is the unchanged picture. Figure 12b shows the scan path which resulted when the participant viewed the scene freely. The scan path in Figure 12c is from when the participant was asked to determine what everyone had been doing just before the visitor arrived. When asked to remember the positions of objects and people the scan path in Figure 12d emerged. In Figure 12e, the participant was asked to determine the ages of all of the people in the scene. Figure 12f was the scan path when the participant was asked to remember what everyone in the scene was wearing. The scan path in Figure 12g emerged when the participant was asked to judge how long the visitor had been away. The task-dependent nature of these scan paths means that it is very difficult to determine what region of a scene is the focus of attention and thus displayed in high resolution based solely on intrinsic properties of a scene.
Figure 12. Differing Scan Paths by Task
(Yarbus, 1967)
1.3. Level of Detail Selection Techniques

Two techniques can be used separately or combined to select an appropriate level of detail, reactive level of detail selection and predictive level of detail selection. Reactive level of detail selection uses information that is provided while the application is running to determine the appropriate level. Predictive level of detail selection uses the properties of the object and scene to determine when to render a specific level of detail before the application is run.

While the studies reviewed in this section are organized by their method of level of detail selection, they vary on other characteristics as well. These include the method of tracking, representation, basis for reduction and display type. Either head or eye tracking is used to determine where the user is looking. The studies presented use discrete, continuous or view-dependent representations. The method of reduction is either screen-based or model-based.

Screen-based level of detail reduction seeks to reduce the bottleneck in the rasterization portion of the graphics pipeline. This type of technique is most useful for teleconferencing, where the original source is a 2D image. Model-based level of detail reduction is for 3D content and involves reducing the geometric complexity of the models in the scene to reduce the bottleneck in the vertex manipulation portion of the graphics pipeline. These selection techniques have been demonstrated on monitors, head-mounted displays (HMDs), and CAVEs.

1.3.1. Reactive Level of Detail Selection

Reactive level of detail selection has been employed most frequently. Reactive level of detail selection methods utilize a direction of view or point of gaze in order to select the appropriate level of detail. This is because when we are attending to things, we look at them. The earliest applications that employed reactive level of detail selection were flight
simulators which used distance from the viewer within the view frustum to determine the level of detail. Figure 13 shows a diagram of a top-down view of the scene illustrating the regions of a scene that would be rendered at various levels of detail using this technique.

Distance-based reactive level of detail selection relies upon the fact that objects that are farther away are rendered smaller on the screen. All that is needed to calculate the level of detail is the distance between the user and the object. This works well for head-mounted displays, because the field of view is relatively small. Figure 14 shows an implementation of distance-based level of detail selection. In this figure, the bunny closest to the viewer is rendered with the highest detail. The middle bunny is rendered at a medium level of detail. The furthest bunny in the scene is rendered at the lowest level of detail. In the image on the top, the distances have been compressed to show the three levels of detail. The image on the bottom shows actual distances to illustrate the difficulty in distinguishing between the appearances of the three levels of detail.
Figure 14. Implementation of Distance-Based LOD Selection
In eccentricity-based level of detail selection, objects in a direction away from the direction of the gaze will be rendered in the lowest detail. The level of detail is determined by an object’s angular distance from the point of gaze. Figure 15 shows a diagram illustrating the regions of a scene which would be rendered at each level of detail using eccentricity-based level of detail selection using a top-down view of the scene.

Note: Gaze direction is from the center to the left

Figure 15. Top-Down Schematic Representation of Eccentricity-Based LOD Selection
An implementation of eccentricity-based level of detail selection is shown in Figure 16. In the top image, the user’s gaze is aimed directly at the bunny. It is therefore rendered in high detail. In the bottom image, the user’s gaze is aimed slightly away from the bunny, causing the model to be rendered at a lower level of detail. In order to determine the appropriate level of detail for eccentricity level of detail selection, the point of gaze must be determined either using head tracking or eye tracking. Because head movements closely follow eye movements, head tracking can be sufficient for determining the center of gaze. This is an easier method to apply given the high cost and awkwardness of eye trackers. The disadvantage of using head tracking over eye tracking is that you will have to preserve a larger region of high detail because you will be uncertain of the exact point of gaze (Watson, Walker, & Hodges, 1997). With eye tracking, the user’s point of gaze is pinpointed exactly, and the degradation of rendered detail can correlate precisely with the reduction of visual sensitivity in the periphery.
Figure 16. Implementation of Eccentricity-Based LOD Selection

a) Looking at the object

b) Looking away from the object
Loschky and McConkie (1999) created an eccentricity-based system that used eye tracking and a monitor for display. Their reduction method was screen-based involving a high resolution area and a low resolution area. They set out to determine the earliest time after an eye movement at which a change in the display could be detected. Visual sensitivity is known to be greatly reduced during eye movements in order to suppress image blur caused by the movement. They used monochromatic photographic scenes and discovered that any change to the image must be made within 12 ms of the end of the eye movement. In their study, their display took 7 ms to change an image, therefore the image change needed to be initiated within 5 ms of the end of an eye movement. Saccade lengths were much shorter when the high resolution portion of the image was only 2 degrees. This is important because when saccades are shorter, more eye movements must be made to perceive the entire scene. It was also found that perceivable changes such as delays of 15 ms or more severe degradation did not have a large impact on task performance (Loschky & McConkie, 1999). A later study showed that update delays of 45 ms do not impact the time required to find an object, but do affect fixation durations (Loschky & McConkie, 2000).

Geisler, Perry and Najemnik (2006) used eye tracking to find gaze location in their study. Their screen-based method involved 6-7 discrete levels displayed on a monitor. They provided a 2D map which defined the appropriate resolution for every portion of a video image based upon its eccentricity and direction from gaze. Blur was undetectable when the region of reduction was 6 degrees.

Reingold and Loschky (2002) tested to see if creating a smooth boundary between the two levels of detail impacted performance. Their reduction method was also screen-based and was displayed on a monitor. They observed no significant difference between performance when there was a smooth boundary between the two levels of detail and performance when the boundary between the two levels of detail was sharp.
Murphy and Duchowski (2001) created a virtual environment with a single multi-resolution object. They were trying to determine the optimal view-dependent level of detail for that object. Participants used a head mounted display and their eyes were tracked. The model was originally rendered at a perceivably low level of detail. The level of detail was gradually increased for different sections of the model, until the participant was no longer able to perceive any change in detail. After finding the “optimal” level of detail based on perception, frame rates were compared to discover the performance improvement of using gaze-contingent rendering. When rendering 24 meshes of the Igea model in the same scene, the best performance improvement was observed. If all 24 meshes were rendered at full detail, frame rates were too slow to measure. Employing gaze-contingency improved frame rates to 20 to 30 frames per second.

Watson, Walker, Hodges, and Worden (1997) used a HMD with a fixed inset relative to the display screen creating two discrete levels of detail. In order to bring an object into focus, the user moved his head. The paradigm is like “looking through a fogged glass mask that has been wiped clean in its center” (Watson, Walker, Hodges, & Worden, 1997). Inset displays performed better than displays of uniformly low resolution and did not perform significantly worse than uniformly high resolution displays (Watson, Walker, & Hodges, 1995). The successful substitution of head tracking for eye tracking indicates that eye tracking is not necessary when the inset is not extremely small. A later study showed that using head tracking with a 45 degree inset can replace eye tracking (Watson, Walker, & Hodges, 1997). Even though eye movements can be made outside of the 45 degree window, if a new fixation point is selected more than 15 degrees away from the original starting point, a combination of head and eye movements is more likely to be used.
1.3.2. Predictive Level of Detail Selection

While reactive level of detail selection methods directly find the center of gaze, predictive level of detail selection methods seek to determine ahead of time what will attract our focus. Since what attracts our attention is most likely to also be the focus of our gaze, the level of detail can be selected based on how likely it is for each object in the environment to attract attention.

Minakawa, Moriya, Yamasaki, and Takeda (2001) created a system using a motion ride—a chair with hydraulics which gives the user a sense of motion—in a CAVE. The motion ride limited the user’s movements and served as a proxy for a head tracker. Because the position in the environment was always known, the experimenters were able to pre-render all of the images. The images presented in this experiment were from a 2D source, and thus a screen-based reduction method was used. Extra detail was provided on the front screen of the CAVE with two extra projectors. The ride apparatus limited the user’s range of motion; therefore the front wall was the most likely target for attention. The system employed two discrete levels of detail. A section in the middle of front wall was rendered with a high resolution and the rest of the walls were rendered at a uniformly lower resolution. No information about perceptual benefits was provided.

Certain features make an object more likely to be the focus of attention. These features include brighter colors, movement, task importance, location in the foreground, inclusion of patterns or faces, and familiarity to the user (Luebke, Reddy, Cohen, Varshney, Watson, & Huebner, 2003). By calculating the influence of these features you can create a saliency map of the most important objects in the environment. A saliency map represents the visual importance of each portion of the visual field topographically. The color of a region indicates how important that region is and how likely it will be to attract attention. For predictive level of detail selection, the objects with the most saliency should be rendered in the highest detail.
Another feature that can be used to predictively select the appropriate level of detail is motion. While motion does attract attention, we also do not have as much time to focus on the details of objects that are quickly moving across the screen. This means that we can render fast-moving objects at a lower level of detail than slower-moving or still objects in the same environment.

Parkhurst and Niebur (2004) employed a velocity-based predictive level of detail selection technique. A model-based reduction technique was used and the display was a monitor. The models used a continuous level of detail representation. All objects in the scene were stationary, so the velocity was based upon the motion created by the rotation of the viewport. The rotation of the viewport caused a reduction in the level of detail of objects in the scene. As the rotation slowed, the level of detail of the objects increased. The time it took to search for objects in the scene was not affected by the level of detail reduction, but the highly variable frame rates did make it more difficult to localize the target objects.

1.3.3. Summary

Table 2 provides a breakdown of the key differences in the experiments described in this section 1.3. Most of the described experiments employed eye tracking and screen-based reduction of 2D images. Discrete representations of the levels of detail were created selection the appropriate level of detail was eccentricity-based. Most of these rendering systems were implemented on single monitor displays.
Parkhurst and Niebur (2002) provided some considerations for evaluating the feasibility of a variable resolution display. A conservative estimate of the visual field is 180 horizontal degrees by 100 vertical degrees. Adaptive rendering has the potential to improve rendering performance more when using a head mounted display than on a computer monitor because it has approximately double the field of view. Actual improvements on rendering performance are also a function of eye (or head) tracker sampling rates, software drawing routing, and display refresh rate.

### 1.4. Immersive Virtual Reality Environments

Head mounted displays and CAVEs present a larger field of view than desktop displays. The limited field of view of a desktop display can cause the user to collide with walls and
doorways in virtual environments (Duh, Lin, Kenyon, Parker, & Furness, 2002). This is because it interferes with the creation of a spatial map of scene and thus make navigation more difficult. Immersive virtual reality systems are able to provide the user with an environment more consistent with reality and therefore do not hinder navigation as much.

Displays with a limited field of view also limit the number of items that can be simultaneously displayed (Tory & Möller, 2004). If more items are included, all of the items must be displayed in less detail in order to fit in the visual scene space. On the other hand, reducing the number of items makes it harder to see the relationship between all of the items. The user must either keep more detail or context information in working memory. The extra cognitive load could adversely affect performance. Displays which provide a larger field of view alleviate the demand for visual scene space, but creating large immersive displays also requires rendering more vertices and pixels.

The method of displaying a virtual environment greatly impacts how a user can interact with the environment. Because so little of the visual input that we perceive is in high resolution, we can work to render each piece of the environment at the lowest level of detail that is perceptually indistinguishable from its non-degraded counterpart. For fully immersive environments, an eccentricity-based level of detail selection is better because it takes into consideration our limited horizontal field of view. In an immersive environment, if the level of detail were selected simply based on distance, much of the scene would needlessly be rendered in high detail. Using distance-based level of detail selection, an object located behind the user would be rendered in high detail because the distance from user to object is small. By employing eccentricity-based level of detail selection instead, that object would be rendered more appropriately in low detail.
1.5. **Research Approach**

The purpose of this research was to determine if perceptually adaptive rendering techniques can improve the quality in an immersive virtual environment. Perceptually adaptive rendering has the potential to improve frame rates while maintaining the detail needed by the user to perform tasks in virtual reality. The increased interactivity provided by faster frame rates will hopefully counterbalance the reduced peripheral detail to improve the user’s overall experience of immersive virtual reality.

The first component of this research was the design of a new perceptually adaptive rendering system which is capable of running on immersive virtual reality hardware. The rendering system supports rapid authoring of virtual environments. This allows for the construction and storage of multiple unique environments. These environments can be displayed without recompiling the source code. Level of detail can be determined using an eccentricity-based technique. The eccentricity-based technique can be combined with an attentional hysteresis factor to prevent objects located at the first threshold angle from switching back and forth repeatedly. Attentional hysteresis is explained further in section 2.2.2.

The second component of this research is the evaluation of the perceptually adaptive rendering system. In order for the system to be useful, it must increase interactivity by providing higher frame rates than if the environment were rendered in uniformly high detail. Also, the changing level of detail must not create extra distraction for the user. In order to validate the system based on these qualifications, a full study and a pilot study were performed.

The experimental approach in this thesis is motivated by prior research on perceptually adaptive displays (Parkhurst & Niebur, 2004). This study used a realistic living room environment, shown in Figure 17. The environment was rendered on a desktop monitor using the Unreal engine. Objects in the scene were represented using continuous
level of detail. The appropriate level of detail was determined based on eccentricity, with an eye tracker determining the participant’s gaze location. An object was considered found when the user moved a crosshair over the correct target object. Although it took longer to localize target objects, none of the participants reported anything unusual about the environment even when prompted.

Parkhurst and Niebur showed that perceptually adaptive rendering could be beneficial even in the limited screen space of a desktop monitor. The research in this thesis focuses on using similar techniques in an immersive environment where the potential benefits are even greater. Discrete levels of detail will be used because no existing virtual reality software
supporting continuous level of detail was found which could perform fast enough. Because head tracking is already provided in the CAVE, head tracking is used instead of eye tracking.

This thesis focuses on the design and evaluation of a perceptually adaptive rendering system using three discrete levels of detail in a CAVE. Chapter 2 describes the rendering system itself which was designed to incorporate perceptually adaptive rendering techniques. Chapter 2 also includes a description of how the rendering system enables the easy creation of immersive virtual environments. The software and hardware used in the rendering system are also described. Chapter 3 outlines the pilot study used to evaluate the rendering system. The pilot study placed more emphasis on the user’s qualitative experience in the virtual environment to find conditions where perceptually adaptive rendering performed the best. It also provided an opportunity to solve any issues with the environment before the full study. The full study is presented in chapter 4. The full study was designed with a larger emphasis on quantitative data. In the full study, the two best perceptually adaptive rendering conditions found in the pilot study were tested more rigorously. Chapter 5 describes overall conclusions and future directions for this research.
CHAPTER 2.  THE RENDERING SYSTEM

This chapter describes the rendering system that was created to implement perceptually adaptive rendering in an immersive virtual reality environment. It was designed to be flexible so it could be used to perform many different tasks. This flexibility extends to the rendering platform, the types of environments which can be rendered, levels of interaction with the environment, as well as the method by which level of detail changing decisions are made. The flexible nature of the application architecture also allows for the rapid authoring of virtual environments which can be stored and reloaded at a later date.

2.1. Interaction Modes

The application can be run in one of four different interaction modes. Each interaction mode was created in order to allow the user to perform a different task in the virtual environment. The interaction mode to be used is specified in a master XML file. Although the user must restart the application in order to switch interaction modes, it is not necessary to recompile any portion of the code. Figure 18 shows the four interaction modes for the application: simulation mode, navigation mode, edit mode, and experiment mode. Simulation mode plays scripted actions in the virtual environment like a movie. Navigation mode allows the user to move around in the virtual environment using a game pad. Edit mode is used to help with virtual environment authoring. Experiment mode was created to test the perceptually adaptive rendering system.
2.1.1. Simulation Mode

Simulation mode was designed around a movie paradigm. The user has no interaction with the application once begins other than to walk around the environment within the confines of the 10x10x10 space defined by the C4 or the C6. This mode is used for a project called Addiction Education Using Science and Technology (AEST) which is intended to teach high school students about the effects of methamphetamine use on human neurology. A frame from this simulation is shown in Figure 19. It depicts the organelles inside a neuron (on the left) and a regular cell (on the right).

The simulation mode has the advantage over a standard movie because the environment can be rendered in 3D in an immersive environment which can lead to a greater feeling of presence. In simulation mode, scene changes, camera movements, model animations, and sound triggers occur automatically based upon the current time in the simulation. In order to aid with scripting longer simulations, the times provided for sound, camera, and model animation events are based on scene time instead of over all simulation time. This means that adding events or time in earlier scenes does not require any changes to the scripting of events in later scenes.
2.1.2. Exploration

Exploration mode allows for interaction with the environment by adding gamepad navigation. With the help of a gamepad, users can change the position and orientation of the viewport. Instead of being a passive observer in the environment, the user is able to explore the environment at his or her own pace. Navigation mode is also useful when the scene creator wishes to ensure that all of the models are loaded and placed correctly in the environment.

When the application is running in navigation mode, script based events are ignored. Scene changes occur as the user desires when designated buttons are pressed. When the user is viewing the last scene defined in the file, pressing the forward scene button causes the first scene in the file to be rendered. When the user is viewing the first scene in the file, pressing
the back scene button causes the last scene defined in the file to be rendered. The velocity of navigation can be manipulated using the throttle on the back of the gamepad. Figure 20 shows the functions that are available when the application is running in exploration mode.

![Figure 20. Available Exploration Mode Interactions](image)

### 2.1.3. Edit

Edit mode allows for the highest level of interaction. The intent of edit mode is to allow a person with minimal coding experience to create and animate his or her own virtual environment. Like exploration mode, in edit mode, the user can navigate around the environment using the game pad. But unlike in exploration mode, in edit mode, the user is also able to select and move objects in the environment and save them out to a script file. Models can be declared in an XML file with minimal information, and then while using edit mode, the objects can be arranged in the environment. After saving the file, edited objects
will automatically appear in their new locations upon restarting the application. Edit mode is actually divided into multiple sub-modes. These sub-modes include a navigation mode, a grabbing mode, and a light manipulation mode. Figure 21 shows the interaction functions for edit mode.

Navigation mode provides the same functionality as the main navigation mode. This mode allows the user to move around in the environment. The only difference between the edit navigation mode and the general navigation mode is the presence of a pointer appearing as a yellow sphere in the middle of the viewport. When the pointer intersects an object, and the user presses a button, the position and orientation of that object is printed in the command line. If the button is pressed when the pointer is not intersecting the object, then the positions and orientations of all grabbable objects are reported in a text console. Pressing another button displays the current camera position and rotation. This feedback is necessary for the creation of animation scripts which will be described in section 2.4.

If an object is intersected by the pointer and a different button is pressed, the application goes into grabbing mode. Once in grabbing mode, the user is able to rotate and position the intersected object. Pressing the button again will send the application back to edit navigation mode. If physics is enabled, then the object will drop down to rest on the nearest flat surface in the same way it would fall if it were dropped in real life. If physics is not enabled, the object will remain in the position and orientation of its release.

It is also possible to enter light editing mode from the edit navigation mode. Light editing mode allows you to add and subtract lights in the scene. The positions and orientations of existing lights may also be modified as well as the intensity and spread of the light.
Figure 21. Available Interactions for Edit Mode
2.1.4. Experiment

Experiment mode was specifically created to carry out the rendering system validation experiments that are outlined in chapters 3 and 4. Participant input via the gamepad is required to advance the experiment. Between trials, experimenter input via the wand is also required to continue. Figure 22 shows the interaction functions for experiment mode. In experiment mode, the paradigm for defining when to switch levels of detail can be changed while the application is running.

![Available Interactions in Experiment Mode](image)

2.2. Level of Detail Selection

Two different factors are used by the rendering system to determine the appropriate level of detail to display. These factors are eccentricity from the user’s head position and attentional hysteresis.

2.2.1. Eccentricity

An eccentricity-based level of detail selection as described in section 1.3.1 is used. Because 3 discrete levels of detail are defined, there are two threshold values: $\theta_1$ and $\theta_2$. The first threshold value, $\theta_1$, is the angle at which the switch between the high and medium level of detail occurs. The second threshold value, $\theta_2$, is the angle which determines the switch between medium and low level of detail. Figure 23 shows the two theta threshold values on a top-down schematic representation of eccentricity-based level of detail. The lower-detail objects are not eliminated completely, because some information about these objects is
encoded for use in choosing the next focus for our gaze (van Diepen & d'Ydewalle, 2003). Also, removing the objects from the scene may cause the luminance to vary too much and reduce the feeling of immersion in the virtual environment.

2.2.2. Attentional Hysteresis

Hysteresis is a lag which is introduced into the switch between two levels of detail. Its purpose is to reduce the oscillating effect which can arise when objects are located on the
border between two levels of detail (Luebke, Reddy, Cohen, Varshney, Watson, & Huebner, 2003). Typically this is implemented by having different threshold distances depending on whether a switch is occurring from a high to a low level of detail or if it is occurring from a low to a high level of detail. By having different threshold distances for both switching directions, the switch will occur later than normal when switching from high to low level of detail and when switching from low to high level of detail.

Instead of using a normal distance threshold hysteresis, this rendering system uses an attentional threshold hysteresis. Our attention is limited. Even if we see something, if it is not attended, then we are not likely to remember it (Rock & Guttman, 1981). Attention is also spatially focused (Posner, 1980). In Posner’s experiment, participants had to respond to a light turning on in either the right or left side of the field of view as quickly as possible. Before the light turned on, a cue appeared in the center of their field of view. The cue would do one of three things: correctly indicate where the light would appear, give the opposite location, or give no information about the location of the light. When the cue was valid, participants were faster to respond to the light turning on. When the cue was invalid, participants were slower to respond to the light. The cue directed participants’ attention to a specific region. If they were attending the correct location, they could respond faster. It is only necessary to render in high detail what is being attended. Since our visual attention is limited to a specific spatial location, attentional hysteresis involves rendering recently attended objects and the objects nearby longer than eccentricity-based level of detail selection would dictate. This means that the changes in level of detail will occur outside of the user’s attended area.

If an object is no longer in the region defined to be the high level of detail, then a counter is started. If the attentional hysteresis threshold, t, is reached, then the object switches to the medium level of detail. The threshold is defined in terms of seconds. If the object returns to the high level of detail region, then the counter is returned to zero. The
addition of an attentional hysteresis factor means that objects remain at a higher detail longer. They only switch to the lower level of detail after they are no longer near the focus of attention. Since the objects are not near the focus of attention, the switch from high detail to medium detail should not be noticeable to the user.

2.3. Using Models

Models can be obtained from wherever available or created using modeling software. In order to be used by the rendering system, they must be converted into an appropriate file format. The following two sections describe the sources for the models used in the validation experiments as well as how the discrete levels of detail of these models were created.

2.3.1. Model Acquisition

Models were acquired from Turbo Squid (Turbo Squid, Inc., 2007) and The 3d Studio (The3dStudio.com, Inc., 2007). Some of the larger polygon count models were acquired from Georgia Tech’s Large Geometric Models Archive (Turk & Mullins, Large Geometric Models Archive) and Stanford’s 3D Scanning Repository (The Stanford 3D Scanning Repository, 2007). Other models were created using a NextEngine’s 3D Desktop Scanner (NextEngine, Inc., 2007).

2.3.2. Model Simplification

The rendering system uses 3 discrete levels of geometric detail. Appendix B shows all of the models used in the pilot and full studies at their three levels of detail. Model simplification is performed offline using QSlim and MeshDev.

QSlim (Garland, 2006), created by Michael Garland, uses a quadric error metric to reduce the error of the mesh while simplifying. The algorithm used in QSlim is capable of simplifying models with or without holes.
Because the command line application simplifies to a specified number of vertices and not a specified error, MeshDev is used to check the error levels among all of the models in the scene. MeshDev compares two triangular meshes and calculates their differences using the attribute deviation metric (Roy, 2005). The unsimplified model is used as the reference model. For each point on the reference mesh, the nearest neighbors to that point on the simplified mesh are found.

It would be desirable to find a solution which uses a continuous level of detail representation or a progressive mesh. OpenMesh (Computer Graphics & Multimedia Group, 2005) was evaluated for use because of its reported compatibility with OpenSG, but level of detail switching rates were not fast enough for use in a real-time rendering system. No other freely available alternative was found.

2.4. XML Scripting

XML, a markup language whose tags make the code resemble HTML, was used heavily to allow environments to be constructed once, and reused many times without having to recompile the code. The XML file format is relatively human legible and there is software which eases the parsing of XML files. The use of XML scripting allows for more flexibility when designing environments or running the application. The XML files used in the rendering system define how the application runs, what the user can interact with, and what and when models, sounds, and movements are used. The following sections describe the various XML files and the attributes which are defined in each.

2.4.1. Master Configuration

The master configuration file is provided in the command line when the application is executed. It provides information that the application needs as soon as it begins running. In this file, the interaction mode is defined. The master configuration file also specifies the file that contains all of the geometry in the virtual environment. If the application is running in
simulation mode, it is useful to declare a simulation duration. When the application time reaches this simulation duration, then the simulation time resets and all of the components set their scripts back to the beginning. Thus, the simulation can run on an infinite loop. A time increment can also be declared. This is useful when saving each frame to a VRML file for later use in a ray tracing application so that constant frame rates for export are maintained even though it takes more time to save scenes with more vertices. For experiment mode, a participant ID is defined. This ID is then used for the log file that is generated for each participant. The structure of this file is illustrated in Figure 24.

![Diagram](image.png)

**Figure 24. Structure of the XML Master Configuration File**

### 2.4.2. Scene Graph Configuration

The scene graph configuration file contains information about all of the rendered objects in the environment. Most aspects of the scene are defined in the scene graph configuration file. This can be seen in Figure 25. The level of detail schema, pointer and skeleton are defined globally for the entire application. Most of the information is stored within the scene. The skybox is used to give the appearance that the scene is sitting in a large world. While the skeleton is defined globally, the skeleton script is defined per scene. The differences between skeleton configuration file and skeleton script will be described in later sections. The actual geometry for the architecture is defined in a different file, but the transformation (position and orientation) of the architecture portion is defined in the scene graph. The presence of the gravity node within the scene indicates that physics will be used in this scene.
The model portion of the XML scene graph configuration file consists of a physics portion and a geometry portion. If gravity is defined, then each model must also have a physics file defined. If gravity is not defined, then the physics node of the xml file will be responsible for describing the object transformations. The geometry node defines the file location for the model to be loaded. If a shader will be applied to the model, then it will be defined here along with the parameters associated with that shader. Currently a simple color shader or a more complex wood texture shader can be selected. The file location of the model script is also defined in the scene graph XML file.

2.4.3. Architecture Configuration

Architecture configuration files define a coherent unit of architecture. This could be a floor, a room, or an entire building. It is useful to define only a single room per architecture file, because then room architecture can be reused and combined to create an entire building. As shown in the explanation of the scene graph configuration file, each architecture unit is given its own position and orientation. The structure of the architecture file is shown in Figure 26.

Static models such as doors and windows can be defined in the architecture configuration file. The format is the same as for normal models in the scene graph configuration file. The unique portion of the architecture XML file is the ability to use primitive box shapes to create surface. Each surface requires the definition of a material to be applied to it. A material is defined by the ambient, diffuse, and specular colors as well as a shininess factor.

Surfaces are either whole or pieced. A whole wall is defined by giving the dimensions of the box and its transformation. A pieced surface is one which has holes in it to provide for doors and windows, but should be transformed together like a whole surface. A
pieced surface is given an overall transformation. Then the smaller pieces are defined with their own dimensions and transforms.

2.4.4. Light Configuration

The light configuration file defines all of the lights in a single scene. Lights can either be spot lights or point lights. The ambient, diffuse and specular components of the light’s color can be specified. The translation and rotation of the light can also be defined. If the light is a spotlight, the spotlight’s exponent and angular cut off can be specified in this file as well. Figure 27 shows the structure of the light configuration file.
Figure 25. Structure of the XML Scene Graph Configuration File
Figure 26. Structure of the XML Architecture File

Figure 27. Structure of the XML Light Object File
2.4.5. Camera Configuration

The camera configuration file defines when and where the camera moves in simulation or experiment mode. There is no explicit camera. Instead, movement is provided by transforming the top of the scene graph hierarchy. The term “camera” is used because it easier to conceptualize. The movements, though provided as if they are camera movements, are transformed within the code to properly move the scene node. Figure 28 shows the structure of the camera configuration file.

The file consists of a series of movements. Each movement has a starting time and duration. Minimally, a destination location must be defined. Locations consist of a combination of position, orientation and scale. An origin location can also be defined. If an origin is not defined, then the origin is taken to be the current transformation of the scene. If this is also not known, then the origin is defined as an identity matrix.

2.4.6. Sound Configuration

The sound configuration file includes all of the sounds which will be played in a single scene. Sounds are used in experiment mode to indicate when a participant has found
the target object and when all trials have been completed. Each sound is defined with the name of the sound and a starting time for the sound. The file location is defined in a different VR Juggler specific configuration file. Figure 29 shows the structure of the sound configuration file.

![SoundConfig diagram]

Figure 29. Structure of the XML Sound Configuration File

2.4.7. Skeleton Configuration

If the scene creator wishes to add an animated skeleton to the environment, a skeleton configuration file must be specified. This file defines all of the bone models which are part of the skeleton, specifies the transforms which are applied to the different bones to position them correctly, and sets the trackers which are associated with each bone. The skeleton configuration file is defined once in the scene graph file even though the skeleton is shared among all of the scenes. The structure of the skeleton configuration file is shown in Figure 30.

2.4.8. Script Configuration

Script files define how the models in the environment behave. The script file is divided into different types of events. Each event has a starting time and duration. The five types of events are placeholders, fades, warps, movements, and animations. Figure 31 shows the structure of the model script file. Model movement events are defined the same way that camera movements are defined. For further information, see section 2.4.5.
Placeholders were created for use in experiment mode where user input and not times determine when specific actions should occur. Placeholders are used when the application needs to advance to allow some models to update, but no action should yet be performed on the current model.

Fades allow objects to disappear and reappear in the scene without loading multiple versions of the same model. Fade events are either fades in or fades out. Although it could be possible at a later date to create a shader which changes the transparency level of an object. Currently, fades involve attaching the model to the scene, fade in, or detaching the model from the scene, fade out.

Warps are similar to movements, but instead of gradually changing the position based upon progress through the movement, a warp is an instant change in location. Warps can also define a new color for a model. When the object appears in its new location, it will also have a different appearance.
Animations call a log file which specifies how the bones in the skeleton will move. New locations can be defined for the skeleton. These locations are applied as warps. The other component to the animation event is the location of the animation file. The animation file has all of the transforms for all of the nodes so the skeleton can perform a specific animation. Once an animation has completed, the animation file will close. If a previous animation has not yet completed, the skeleton class will cancel the previous animation, close its file, and open the new animation’s file.

2.4.9. Level of Detail Schema Configuration

The level of detail schema configuration file allows the method of determining levels of detail to change without restarting the application. This is necessary for the experiments described in chapters 3 and 4. The structure of the level of detail schema file is shown in Figure 32. If level of detail management is used, then at least one schema must be defined in this file. The schema will stay in effect for the specified number of iterations. These iterations are equivalent to the number of times an object is found in the search task used in the pilot study described in chapter 3. A starting level of detail is also defined. This is useful for cases when uniformly high, medium, or low detail should be used. The two factors for the level of detail schema are eccentricity and hysteresis. When using eccentricity, the threshold for switching between high and medium detail and the threshold for switching between medium detail and low detail must be defined. When hysteresis is used, then the delay value in seconds must be defined.
Figure 31. Structure of the XML Script File

Figure 32. Structure of the XML LOD Schema File
2.5.  Code Structure

The two main portions of an interactive virtual environment are the actions that can be taken in the environment and the objects with which a person can interact. This section describes how the Command design pattern is used to handle actions and the scene graph which maintains the objects to be rendered.

2.5.1. Command Design Pattern

The rendering system uses the Command design pattern (Gamma, Helm, Johnson, & Vlissides, 2002). The Command design pattern turns actions into objects allowing them to be passed around to different clients. One advantage of the Command pattern is that it is easy to add new commands without modifying all of the other classes. Also, the command creator does not need to have knowledge of the command receiver and vice versa. Both creator and receiver need to agree upon the structure of the commands, so that they can be handled properly.

In a clustered system, it is only possible to plug a game controller into one computer. This makes it more challenging to interact in an immersive virtual environment which uses multiple computers to render the scene. In order to ensure that user interactions are received and applied on all of the computers running the application, the commands are generated on one computer and shared with the rest of the computers. For more information about clusters, see section 2.7.

Commands are stored in a vector and passed to all of the computer nodes each frame. At the beginning of the frame, the different components have a chance to update and add commands to the vector on the master machine. Then that vector is shared with the rest of the computers in the cluster. At the end of the frame, each component running on each computer handles any commands which are pertinent to that component. Figure 33 shows the components which are included in the rendering system. The command list starts in the
SceneCreator component and is passed down the hierarchy. Figure 34 shows each component and the commands that the components accept. Notice that the SceneManager and ModelManager components accept different commands based upon the mode in which the component is currently running.

Figure 33. Hierarchy of Control for the Component Managers
Figure 34. Commands Accepted by the Components in the Rendering System
The components which are children of the Scene: SoundManager, CameraManager, ModelManager, and SkeletonManager take a different command “Scene_Time” instead of the overall application time “Current_Time”. This is to aid in scripting long simulations. If an overall cumulative time was used, adding a scene in the beginning would mean that starting times for all events following the new scene would have to be changed. By using a scene-based time for the children of the scene, adding and subtracting scenes becomes easier. Figure 35 illustrates the concept. The SceneManager requires using the cumulative time because it needs to know when to change scenes.

![Figure 35. Using Scene Time](image)

### 2.5.2. The Scene Graph

A scene graph is a method of organizing the components of a graphical environment in a logical manner. They are represented as a collection of nodes in a tree structure. A parent node may have multiple children, but a child node can only have one parent. If a
parent node is transformed, then all of its children nodes will also be affected. Transforms to a child node do not affect the parent. The best way to understand this structure is to imagine the human skeleton. If you move your upper arm, your lower arm moves too. But, moving your lower arm does not move your upper arm as well. If your skeleton was represented in a scene graph, your lower arm would be stored as a child of your upper arm. In the virtual environment, if the room moves, you want the objects in the room to move as well, but moving an object inside the room should not cause the room to move too.

Figure 36 illustrates how the nodes in the scene graph are arranged. There must be at least one light defined so that the objects in the environment are visible. It is possible to have more than one light, and to have both point lights and spot lights in use at the same time. Although it is not necessary to define architecture, if physics is enabled, a floor must be defined. Otherwise there will be nothing in the scene to stop the objects from falling out of the view frustum. Although not shown in Figure 36, the architecture node can also hold permanent models such as windows, doors or anything else which will not move when other objects in the scene collide with it. The bone nodes in the skeleton are defined the same way as the model nodes.
2.6. Software Libraries

The main software libraries used in the rendering system are VR Juggler, OpenSG, libxml2, OPAL, and ODE. These libraries and their application are explained in this section.

2.6.1. VR Juggler

VR Juggler (http://www.vrjuggler.org) is an open source virtual reality application platform which allows a program to run on a variety of virtual reality systems without having to recompile the source code. Using VR Juggler, the same piece of code can be used to display a virtual environment on a desktop computer, in a CAVE, or with a head-mounted
display. It handles distributing applications across multiple machines for environments like
the CAVE where at least one computer is used for rendering each screen in the system. VR
Juggler is also responsible for handling input devices as sharing the data from those devices
across multiple computer nodes. Representation of core math data types used in graphics
applications such as matrices, vectors and quaternions are provided in VR Juggler by the
Generic Math Template Library or GMTL (http://ggt.sourceforge.net/).

The Sonix module of VR Juggler is also employed in the rendering system. Sonix is
a sound abstraction layer which can be used as a wrapper around various sound libraries such
as OpenAL (www.openal.org/) and Audiere (http://audiere.sourceforge.net/). The rendering
system works with the Audiere sound library.

2.6.2. OpenSG

OpenSG (http://opensg.vrsource.org/trac) is a scene graph system for real-time
graphics applications. It is an abstraction of the lower level OpenGL that makes handling
graphics objects easier for the programmer. OpenSG is used to store all of the objects which
will need to be rendered in the virtual environment. It also allows the application of
advanced shaders to models stored in the scene graph. It was selected because of its
compatibility with VR Juggler.

The main portions of the OpenSG scene graph are called nodes. Each node has a core
of a specific type which determines the behavior of the node and any children of that node.
The most common type of core is a transform core. Transform cores contain a matrix which
holds the transformation (translation, rotation, and scale) of any of the nodes children.
Another possible core type is geometry. Geometry cores contain the vertices that define
shapes or models. Possible light cores are point lights, spot lights, and directional lights. I
make use of point and spot lights in my rendering system. Point lights are most commonly
used and involve simpler calculations. They require the definition of a light position. Spot
lights are more computationally expensive to use, but create more realistically lit environments. In addition to a position, spot lights also require the definition of a direction as well as an exponent value which sets the power of the spot light and a cut off angle which sets how wide the light spreads from its point source.

Because the OpenGL Shading Language (GLSL) is employed to create shaders in the rendering system, the application runs best on graphics cards with the OpenGL 2.0 core. The application will run on computers with older graphics cards without GLSL, but some of the materials will not render correctly and performance can be degraded. The rendering system also runs best on computers running Linux, although it can function on a computer running Windows XP.

2.6.3. Libxml2

As illustrated in section 2.4, XML is used heavily in the rendering system to create unique and display unique environments without recompiling the source code. XML handling is performed using libxml2 (http://xmlsoft.org/index.html). Libxml2 is the C parser and toolkit provided by GNOME desktop environment, but it can be used outside of GNOME. It provides functions which allow reading data from XML files as well as changing data within the XML tree structure and saving the changes to be used later.

2.6.4. OPAL and ODE

Physics simulation allows for easier placement of objects in the environment. When objects are placed where the user desires, they can be released and physics will ensure that the object does not float in the air. Physics is implemented using a combination of the Open Physics Abstraction Layer (OPAL) and the Open Dynamics Engine (ODE). ODE (http://www.ode.org/) is a library for simulating rigid body dynamics and detecting collisions. OPAL (http://opal.sourceforge.net/index.html) is a physics engine API which provides a wrapper around ODE functions using simpler C++ function calls and data objects.
2.7. Hardware Description

The rendering system is designed to be compatible with the C4 and C6 CAVEs housed in the Virtual Reality Application Center (VRAC) at Iowa State University. The rendering system can also be run on desktop computers, although the perceptually adaptive rendering capabilities will not be as useful given the smaller display size. This section describes the hardware and layout of the C4 and the C6 as well as the minimum hardware requirements for running the rendering system.

The C4 and C6 are both powered by a cluster of computers. A cluster is a group of computers that work together very closely to share the computational load on complex tasks. They are usually connected to each other via a fast local area network (LAN). The computers for the C4 and C6 cluster respectively are synchronized so that each rendered frame is displayed only when all computers have finished calculating the rendering. This way, the view moves consistently for all of the walls.

Both the C4 and the C6 are capable of rendering environments in stereo. This gives the environment dimensionality. The system switches between rendering the view for the left eye and the view for the right eye very rapidly. Viewers wear a pair of Stereographics CrystalEyes LCD shutter glasses. These glasses are synced up with the computers. When the computers render the view for the left eye, the shutter glasses block out the right eye. When the computers render the view for the right eye, the glasses block out the left eye. This means that each eye only sees the view that has been created for it. The images are combined in the brain and give the user the impression of a three-dimensional scene.

The C6 is a 10 foot by 10 foot room where all four walls, the floor, and the ceiling are screens capable of displaying back-projected stereoscopic images. Figure 37 shows a rendering of the layout of projectors walls, and mirrors in the original C6. The new configuration of the C6 uses four projectors on each side instead of one. The cluster responsible for rendering in the C6 consists of 48 display nodes, 1 master node and 1 audio
node. The computers are HP xw9300 Workstations with dual nVidia Quadro FX4500 graphics cards. The 24 projectors are Sony SXRD. Each projector requires 4 DVI inputs to generate a 4096 x 4096 pixel rendering. Tracking in the C6 is carried out with an Intersense IS900 VET and there are SoniStrip emitters where the ceiling intersects the front, left, and right walls.

The C4 possesses four rear-projected walls: front, right, left, and floor. The floor projection is from the top instead of from the bottom. The right and left walls are capable of moving so that environments can be shown in the closed configuration with a 12’ by 9’ front, left and right walls and a 12’ by 12’ floor. In the open configuration, there is a 36’ wide by 9’ tall front wall and a 12’ by 12’ floor. Figure 38 shows the application running in the C4 in the closed configuration. The C4 uses an Ascension MotionStar Wireless magnetic tracking system. Four computers are responsible for rendering the virtual environment, one for each wall.
Figure 37. Rendering of the Original C6

Figure 38. Virtual Environment Displayed in the Closed Configuration C4
CHAPTER 3. PILOT STUDY

A pilot study was performed to test the application platform and to determine the best perceptually adaptive rendering parameters for in-depth testing. The pilot study also helped to identify problems with the design of the experiment and the virtual environment so that they could be corrected before running a study.

The motivation of this study was to determine general trends for various perceptually adaptive rendering conditions. Participants performed a visual search for a target object which was presented when they entered the virtual environment. Perceptual responses were gathered and search times, frame rates, and head positions for each frame were collected. It was expected that frame rates would improve as the angle that determined the switch between high and medium detail decreased. It was also expected that employing attentional hysteresis would mask the switch to the lower detail model. While quantitative data was collected, more emphasis was placed on the qualitative responses from the participants about the perceptual quality of the scene.

3.1. Methods

3.1.1. Participants

Participants in the pilot study were drawn from the Spring 2007 Psychology department subject pool at Iowa State University. There were a total of 16 participants. Five of the participants were female and eleven of the participants were male. All had normal or corrected to normal vision and reported not being prone to motion sickness. Participants received extra credit in a psychology course for their participation in the experiment.

3.1.2. Task

Participants were required to search for a target object in a virtual living room environment. The living room contained 30 objects chosen from the models in Appendix B.
There were two possible target objects. The dragon was chosen as a target object because its outline was very different from the other models in the environment. The monkey was chosen as a target object due to its similarity to other models. Participants either received the easy task (dragon) or the hard task (monkey). Figure 39 shows the two target models at their three levels of detail. Some objects in the environment were duplicated, but only one target object was presented at any one time. The color of each of the 30 objects was randomly chosen from six colors: red, green, blue, cyan, magenta, and yellow. Randomly assigning object colors insured that the objects would be found based on their shape, and not due to color. When the target object was found, participants pressed a button on the game pad to stop a timer. They then verbally reported the color of the object. This was required in order to verify that the object had indeed been found.
Figure 39. Models Used as Targets in the Pilot Study at Each of Three Levels of Detail

a) Monkey model  
b) Dragon model
3.1.3. Procedure

The pilot study took place in the C4, a four-sided CAVE environment, at Iowa State University. For more information, see section 2.7. Upon entering the room, the participants were asked to read and sign an informed consent form. All participants were offered a copy of the form. After signing the form, participants were asked to fill out a questionnaire concerning their visual acuity. The questionnaire was taken from Sloan et al. (1992) and can be found in Appendix C. The visual acuity questionnaire evaluates eight different visual abilities including: color discrimination, glare disability, light/dark adaptation, acuity/spatial vision, depth perception, peripheral vision, visual search, and visual processing speed. An example question dealing with peripheral vision is “I have trouble seeing moving objects coming from the side until they are right in front of me.” Responses were on a scale of 1 (never) to 5 (always). This questionnaire was administered in order to understand any discrepancies in the verbal reports of the virtual environment’s perceptual quality.

Upon completion of the visual acuity questionnaire, participants were asked to fill out a simulator sickness questionnaire. The simulator sickness questionnaire was taken from Kennedy et al. (1993) and has been provided in Appendix D. The questionnaire lists various symptoms of simulator sickness such as nausea, fatigue, and eye strain. Participants rated these symptoms on a scale of 0 (none) to 3 (severe). If participants were unclear about any terms on the questionnaire, they were told to ask for clarification. Participants were informed that if they felt any of the conditions listed on the simulator sickness questionnaire during the course of the experiment to an extent that they did not wish to continue, the study would be halted immediately.

After these two preliminary questionnaires were completed, participants were given instructions for the experiment. The first eight participants were given purely verbal instructions and were not shown the environment or the changing level of detail. The first eight participants had difficulty understanding the task from purely verbal instructions. The
second eight participants were brought into a sample virtual environment which included the two rooms and a single object placed in the same location as the target object in the experimental virtual environment. This was done familiarize participants with the virtual environment before they had to perform their tasks. They were shown the object at each level of detail and were told to report if they noticed any detail changes while they were performing the task. This was done to acquint the participants with the virtual environment before the task and to see if awareness of the potential for changing level of detail make participants more likely to notice the changes during the experiment. Participants were given instructions on how and when to use the game pad. The game pad was used to indicate when the user was confident of the shape of the target object, when they were ready to start the search task, and when they had found the target object. They were shown the six potential colors for the target object during the search (red, blue, green, yellow, cyan, or magenta). Participants made a verbal report of the target object color each time it was found. I informed them that I would be sitting on a chair in the back of the C4 to write down the reported colors.

Once the experiment began, the participants had the opportunity to walk around the environment to inspect the target object and to become familiar with the virtual environment. The object could be inspected from three sides. When participants were ready, they were required to stand in the middle of the C4 (marked by tape on the floor) and press the ready button on the gamepad. This would cause the target object to disappear and the viewport to move to the next room. As the viewport moved, participants were asked to verbally name the object for which they would be searching. If they could not identify they object, they were asked to describe its features.

Once in the second room, participants were encouraged to turn their head and look at the layout of the furniture in the room. Participants had to turn their heads to view the entire scene because they were located in the center of the room. The virtual environment
contained approximately 6,600,000 polygons. The scene was rendered in stereo. Figure 40 illustrates the distribution of objects on the four surfaces of the C4. When they were ready, they pressed the button and the objects appeared. Upon finding the target object, participants pressed the gamepad button and verbally reported the color of the object to verify that the correct object had been identified. A cymbal sound played when the participants pressed the button after finding an object so that they knew the button press had been registered by the computer. Pressing the button caused the objects to disappear and reappear in a new location so that the participant could search for the object again.

Participants searched for the object five times. The objects did not appear again and participants were asked a series of questions. These post-task questions are provided in Appendix E. An example question from the post-task questionnaire is “How much did the visual display quality interfere or distract you from performing the assigned task.” The questions were designed to determine if participants noticed anything unusual about the presentation of the objects. The gamepad was disabled at this point so the questions could be asked without the interference of objects in the scene. Once the questions were answered, the experimenter re-enabled the gamepad. A button press on the gamepad would cause objects to appear again but a different experimental condition would be used to determine levels of detail.
Figure 40. Visualization of Object Layout in the Virtual Environment
This task was performed in six different experimental conditions. Upon completion of the 30 searches, a harp sound would play to signal the end of the virtual reality portion of the experiment. Participants were asked to fill out two additional questionnaires. The simulator sickness questionnaire was administered once again to determine how their experience in the virtual environment had affected them physically. The second questionnaire asked about their experience in the virtual reality environment including the responsiveness of the environment and to what the degree the environment felt engaging. This questionnaire is provided in Appendix F. Participants were debriefed and given an opportunity to ask questions.

3.1.4. Experimental Design

There were six different experimental conditions. In the first two conditions, all of the objects were rendered with uniform detail. The first condition presented objects in uniformly high detail. The second condition presented objects in uniformly low detail. The next two conditions used eccentricity to determine the level of detail. These two conditions

Note: Gaze direction is from center to left

*Figure 41. Top-Down Representation of Eccentricity*
manipulated the angle at which the object changed from high to medium detail ($\theta_1$). The angle at which the object changed from medium to low detail ($\theta_2$) remained constant. Figure 41 shows a top-down schematic of the eccentricity based conditions with $\theta_1$ and $\theta_2$ marked. The last two conditions employed both eccentricity and attentional hysteresis to determine the level of detail. See section 2.2 for an explanation of these two parameters. The $\theta_1$ and $\theta_2$ values were the same as the previous two conditions, but a delay ($t$) was added to the switch between high and medium detail so that the object remained at the higher detail level longer than if just eccentricity were used. All participants experienced the experimental conditions in the same order.

The first eight participants were not allowed to view a sample environment before they began their task. After it was discovered that participants had difficulty understanding the task with purely verbal instructions, the second eight participants were shown a sample environment and the level of detail switching technique was demonstrated to them to see if that would increase the chances of them noticing the level of detail switching while performing the task. The first eleven participants performed the search task under the same level of detail switching conditions. This was done to ensure that the application was running properly and to determine difficulties with the experimental design. Table 3 summarizes actual values used for each participant in the pilot study.
Table 3. Experimental Conditions for Participants in the Pilot Study

<table>
<thead>
<tr>
<th>Participant</th>
<th>Target</th>
<th>Condition</th>
<th>Shown Sample</th>
<th>Informed about LOD</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>1</td>
<td>Monkey</td>
<td>High</td>
<td>Low</td>
<td>30°, 90°, 0.0s</td>
</tr>
<tr>
<td>2</td>
<td>Dragon</td>
<td>High</td>
<td>Low</td>
<td>30°, 90°, 0.0s</td>
</tr>
<tr>
<td>3</td>
<td>Monkey</td>
<td>High</td>
<td>Low</td>
<td>30°, 90°, 0.0s</td>
</tr>
<tr>
<td>4</td>
<td>Dragon</td>
<td>High</td>
<td>Low</td>
<td>30°, 90°, 0.0s</td>
</tr>
<tr>
<td>5</td>
<td>Monkey</td>
<td>High</td>
<td>Low</td>
<td>30°, 90°, 0.0s</td>
</tr>
<tr>
<td>6</td>
<td>Dragon</td>
<td>High</td>
<td>Low</td>
<td>30°, 90°, 0.0s</td>
</tr>
<tr>
<td>7</td>
<td>Monkey</td>
<td>High</td>
<td>Low</td>
<td>30°, 90°, 0.0s</td>
</tr>
<tr>
<td>8</td>
<td>Dragon</td>
<td>High</td>
<td>Low</td>
<td>30°, 90°, 0.0s</td>
</tr>
<tr>
<td>9</td>
<td>Monkey</td>
<td>High</td>
<td>Low</td>
<td>30°, 90°, 0.0s</td>
</tr>
<tr>
<td>10</td>
<td>Dragon</td>
<td>High</td>
<td>Low</td>
<td>30°, 90°, 0.0s</td>
</tr>
<tr>
<td>11</td>
<td>Monkey</td>
<td>High</td>
<td>Low</td>
<td>30°, 90°, 0.0s</td>
</tr>
<tr>
<td>12</td>
<td>Dragon</td>
<td>High</td>
<td>Low</td>
<td>20°, 90°, 0.0s</td>
</tr>
<tr>
<td>13</td>
<td>Monkey</td>
<td>High</td>
<td>Low</td>
<td>20°, 90°, 0.0s</td>
</tr>
<tr>
<td>14</td>
<td>Dragon</td>
<td>High</td>
<td>Low</td>
<td>20°, 90°, 0.0s</td>
</tr>
<tr>
<td>15</td>
<td>Monkey</td>
<td>High</td>
<td>Low</td>
<td>20°, 90°, 0.0s</td>
</tr>
<tr>
<td>16</td>
<td>Dragon</td>
<td>High</td>
<td>Low</td>
<td>10°, 90°, 0.0s</td>
</tr>
</tbody>
</table>

The condition format is θ₁, θ₂, t.

* Participant was shown level of detail changes, but was not explicitly told about them.
3.2. Results

3.2.1. Visual Acuity

All participants fell within the normal range for the visual acuity characteristics of depth perception, peripheral vision, visual search and visual processing speed. Participant 11 reported a slight impairment in acuity/spatial vision.

Figure 42 shows a box plot of the participant’s visual acuity scores for each of the eight areas of visual function tested by the visual acuity questionnaire. The highest score for color discrimination was 2, which means that participants rarely had difficulty telling colors apart. Search times should not have been impacted due to inability to decide on the target object’s color. Participants also rarely experienced problems with peripheral vision or visual processing speed, both of which are important in a visual search task. One participant reported often having issues with seeing detail in objects, but that participant’s verbal reports of the environment’s visual quality were not greatly different from the other participants’ reports. The scores on the visual acuity questionnaire indicated no other reason to believe participants would have issues performing the visual search task.
Figure 42. Visual Acuity Scores for Participants in the Pilot Study


3.2.2. Simulator Sickness

In the pilot study, the average change in nausea rating was 14.9, oculomotor impairment rating was 23.2, disorientation rating was 31.3, and overall rating was 25.7. These values are shown in Figure 43. The error bars in Figure 43 are large. This is most likely because simulator sickness can affect people in many different ways. Some people are more susceptible to the symptoms than others.

The ratings for the various symptoms in the pilot study were generally rated as “slight” with only a few participants reporting “moderate” symptoms. One participant exhibited more symptoms of nausea, disorientation, and oculomotor impairment than the other participants. That participant’s overall simulator sickness was in the “moderate” range. All the other participants exhibited symptoms in the “slight” range. With the exception of nausea symptoms, the participants in the pilot study reported higher instances of simulator sickness than the participants in Kennedy et al.’s study (1993). In that paper, the average scores were 7.7 for nausea, 10.6 for oculomotor impairment, 6.4 for disorientation, and 9.8 overall.

The difference between the values, in particular the disorientation scale, probably arises due to the fact that participants in Kennedy et al. were trained fighter pilots and the participants in the present study had little to no experience in virtual environments. Furthermore, participants in the Kennedy et al. study reported experiencing no symptoms before entering the simulator. This was not the case in the present study. This can be seen in Figure 44 which shows total simulator sickness experienced by the participants before and after the experiment. Figure 45 illustrates the nausea, oculomotor, and disorientation symptoms experienced by the participants before and after their experience in the virtual environment. Ratings were collected before and after to ensure that conclusions were based on symptoms caused by the environment and not a pre-existing condition. Overall, simulator sickness symptoms were slight. This indicates that it is not likely dangerous to perform tasks in the virtual environment under any of the experimental conditions.
Figure 43. Comparison of Simulator Sickness Averages
Three of the participants with the highest simulator sickness ratings were in timeslots around noon and five in the evening, both of which are meal times. Greater simulator sickness may have been experienced by participants who had reported having three or more conditions on the visual acuity questionnaire where they had problems “sometimes” or more often. Eliminating participants based on these two criteria, testing time and visual acuity, would improve average simulator sickness to levels equal to or lower than that of Kennedy et al.’s with the exception of disorientation, which would still be slightly higher.

Figure 44. Overall Simulator Sickness in the Pilot Study
Figure 45. Simulator Sickness Ratings in the Pilot Study Separated by Type
3.2.3. Perceptual Measures

Perceptual quality was measured through qualitative and quantitative questions after participants viewed the environment in each condition. Overall quality of the environment was measured in the end-of-the-study questionnaire.

Most participants noticed that in the low detail condition all of the objects lacked detail. In the high detail condition, participants mentioned the fact that the movement of the environment was choppy. Two participants noticed that there was a level of detail change when the switch from high detail to medium detail occurred at 30 degrees eccentricity. In the \( t = 1.0s \) and \( t = 2.0s \) hysteresis conditions, participants noticed that the left side looked clearer than the right side. Many participants reported that objects located at the intersection of the C4 walls or objects on the floor appeared distorted. Table 4 shows some of the statements participants made after performing the search task in the various conditions.

<table>
<thead>
<tr>
<th>Condition</th>
<th>Participant Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>High Detail</td>
<td>&quot;head moving isn't smooth&quot;</td>
</tr>
<tr>
<td></td>
<td>&quot;choppy changing&quot;</td>
</tr>
<tr>
<td>Low Detail</td>
<td>&quot;blurry&quot;</td>
</tr>
<tr>
<td></td>
<td>&quot;objects lack detail&quot;</td>
</tr>
<tr>
<td>( \theta_1=10^\circ, \theta_2=90^\circ, t=0s )</td>
<td>--</td>
</tr>
<tr>
<td>( \theta_1=20^\circ, \theta_2=90^\circ, t=0s )</td>
<td>&quot;some of the objects are moving&quot;</td>
</tr>
<tr>
<td>( \theta_1=30^\circ, \theta_2=90^\circ, t=0s )</td>
<td>&quot;little blurry&quot;</td>
</tr>
<tr>
<td></td>
<td>&quot;feels like things are moving&quot;</td>
</tr>
<tr>
<td>( \theta_1=45^\circ, \theta_2=90^\circ, t=0s )</td>
<td>&quot;looks clearer than the last condition&quot;</td>
</tr>
<tr>
<td>( \theta_1=20^\circ, \theta_2=90^\circ, t=1s )</td>
<td>--</td>
</tr>
<tr>
<td>( \theta_1=30^\circ, \theta_2=90^\circ, t=1s )</td>
<td>--</td>
</tr>
<tr>
<td>( \theta_1=10^\circ, \theta_2=90^\circ, t=2s )</td>
<td>--</td>
</tr>
<tr>
<td>( \theta_1=20^\circ, \theta_2=90^\circ, t=2s )</td>
<td>&quot;the left side looks clearer than the other&quot;</td>
</tr>
<tr>
<td>( \theta_1=30^\circ, \theta_2=90^\circ, t=2s )</td>
<td>&quot;objects on one side are blurrier&quot;</td>
</tr>
<tr>
<td>( \theta_1=45^\circ, \theta_2=90^\circ, t=2s )</td>
<td>&quot;objects on one side look blurry&quot;</td>
</tr>
</tbody>
</table>

No attentional hysteresis was used in conditions where \( t = 0s \).
After each condition, participants were asked the degree to which the visual display quality interfered or distracted them from performing the search task. Figure 46 shows the average response for each condition. The $\theta_1 = 10^\circ$ conditions with either $t = 0s$ or $t = 2s$ had the best ratings, but there was only one participant in each of these conditions. The $\theta_1 = 45^\circ$ condition with $t=0s$ was rated the next best.
Figure 46. Frame Rates, Search Times and Distraction Ratings for the Pilot Study.

(Numbers at the base of the bars indicate number of observations.)
The quantitative perceptual measure of the amount of distraction each condition indicates that the best conditions were the $\theta_1 = 10^\circ$ conditions with $t = 0s$ and $t = 2s$. There was only one participant who viewed these two conditions. That participant rated all of the conditions as a “1” which was “not very distracting” with only the high detail condition numerically rated as a “2”. The 45 degree $\theta_1$ condition with no hysteresis was viewed by eleven participants and had the next best average rating of 1.4. Because more participants viewed this environment and it still had a low rating, this makes this condition more likely to be the best for perceptually adaptive rendering.

Figure 47 shows the average responses for the questions on the end-of-the-study questionnaire. Participants reported not being able to inspect objects very closely or from many viewpoints. This is because objects were small and participants were not able to move their feet when they were in the room. The gamepad was not reported as a distraction during the search task. Participants did not have problems with the system’s responsiveness in general.
Figure 47. Responses to the End-of-the-Study Questionnaire
Participants reported being able to concentrate well on the search task, but they were still moderately aware of events occurring in the real world. Participants also reported that the virtual environment was only moderately consistent with reality. Despite the virtual environments inconsistency with reality, participants had few problems adjusting to virtual reality. The responses on end-of-the-study questionnaire indicate that there are no significant issues with the virtual reality environment.

3.2.4. Performance Measures

This section describes the data collected to measure performance of the rendering system as well as the performance of the participants under the various level of detail conditions. These data included frame rates for each condition, angular distance traveled to find the target, and response time for each search.

The participants only performed five searches under each condition. Because not all of the objects in the environment could be seen simultaneously, the search path the participant took had a large impact on the time a participant took to localize an object. Having more search trials will make the mean response time more reliable in the presence of variability.

Figure 46 compares the average frame rates, search times, and distraction ratings for each of the conditions. While the $t=2s$ hysteresis value did improve frame rates for some of the conditions in the $\theta_1=45^\circ$ condition, frame rates were impaired further than the uniformly high detail condition. This is most likely because participants moved their head so fast that the entire environment was rendered in high detail. The distraction ratings are not very different between the high and low detail conditions. The slow frame rates in the high detail condition cause the display to reflect changes in the user’s viewpoint much slower than they can turn their head. This can be very distracting. This effect equals the distraction caused by the low detail condition where important visual information is left out. The pilot study
measures have been separated by eccentricity $\theta_1$ in Figure 48 and by attentional hysteresis $t$ in Figure 49.

Figure 48 shows that the lowest distraction ratings were given to the $\theta_1 = 10^\circ$ and $45^\circ$ conditions. All of the perceptually adaptive rendering techniques generated faster search times than the uniformly high and low detail conditions, but an increasing or decreasing pattern cannot be established. As expected, frame rates decreased as $\theta_1$ increases because more of the scene remains high detail when the $\theta_1$ is large. The $\theta_1 = 45^\circ$ condition appeared to cause lower frame rates than the uniformly high detail condition, but Figure 46 shows that this is only true for the $t=2$s hysteresis condition. The $t=2$s condition caused the entire environment to be rendered in high detail for much of the time and there was some overhead from switching between levels of detail which did not exist in the uniformly high detail condition. This would explain the lower frame rates than the uniformly high detail condition.

Figure 49 shows that the lowest distraction ratings occurred at the $t=0$s and $2$s conditions, but frame rates for the $t=2$s condition were equal to the frame rates in the uniformly high detail condition. The search times for all three perceptually adaptive conditions were equal. Because neither search times nor frame rates were clearly improved by attentional hysteresis, it seems like the best perceptually adaptive conditions were the ones which only varied level of detail as a function of eccentricity.
Figure 48. Pilot Study Measures Separated by Eccentricity Value
Figure 49. Pilot Study Measures Separated by Hysteresis Value
3.3. Discussion

The purpose of the pilot study was to test the virtual environment and the experimental protocol for use in the full study. The pilot study was also used to identify general trends in performance due to changing eccentricity and attentional hysteresis.

A simulator sickness questionnaire was administered before and after participants experienced the virtual environment. A visual acuity questionnaire was administered at the beginning of the study. Participants performed a visual search task under six different conditions. The first two conditions presented the environment in uniformly high detail and uniformly low detail. The second two conditions tested eccentricity factors. The final two conditions tested attentional hysteresis factors. Participants were asked to describe any differences in the visual quality of the conditions and rate each condition’s level of distraction. Average frame rates were collected for each condition. Search times were also recorded for each participant.

Participants in the pilot study experienced a higher level of simulator sickness than participants in the Kennedy et al. study (1993). None of the symptoms were rated as “severe”, but precautions will be taken in the full study to reduce the simulator sickness ratings. Three of the five participants who experienced more simulator sickness had timeslots during lunch or dinner times. Three of the five participants reported issues “sometimes” or higher on 3 or more visual acuity components. One participant had a timeslot during a meal time and reported issues with visual acuity. This accounts for all five participants in the pilot study with higher simulator sickness ratings. In the full study, timeslots will not be scheduled during meal times. Participants in the full study reporting visual acuity issues “sometimes” or more on three or more conditions will not be allowed to enter the virtual environment.

Search times in the pilot study were longer than expected. Average search times were around fifteen to twenty seconds. There are two reasons why this may be the case. First of
all, objects did not cover a large portion of the screen. Because objects were small, detail was hard to see. Object sizes were dictated by the limitations of some of the furniture in the virtual room. In the full study, height-limiting furniture will be replaced. Object size will be increased so that more detail is visible. Another reason why search times were longer than expected may be because there were too many objects in the room. This made the search task too difficult. The number of objects in the room will be decreased. This will lower the average search times in the full study.

Participants had difficulty noticing the low detail condition while it was happening. After seeing the next condition, they realized that the previous one had been fuzzy. One reason for this may be that for the taller participants, more of the models were rendered on the floor and appeared to be malformed. The smaller object size may have also reduced the amount of detail visible to participants. In the full study, shelves will be added to the walls to place objects higher in the field of view which will make object identification easier. As stated before, the object size will also be increased to make detail more visible.

As expected, frame rates decreased as the $\theta_1$ eccentricity value increased. This is because more of the scene had to be rendered in high detail as $\theta_1$ increased. When attentional hysteresis was not used, all $\theta_1$ values resulted in higher frame rates than the uniformly high detail condition. The uniformly high detail condition allowed frame rates that support interactivity (approximately 20 frames per second). This means that the rendering system is capable of handling more polygons. In the full study, the scene will be modified to contain more polygons.

Attentional hysteresis did not improve the quality of rendering. More of the environment had to stay at high detail. Search times did not appear to significantly change for any of the values of $t$. Distraction ratings were also not reduced. The experimental design for the pilot study may not be optimal for the testing the usefulness of attentional hysteresis. This is because in a visual search task an object will quickly be dismissed when it
does not appear to be target. Once rejected, an object is not likely to be revisited until the other objects in the scene have been inspected. It may also be the case that the hysteresis values were too high. In the full study, eccentricity-only conditions will be tested.

The $\theta_1 = 10^\circ$ and the $\theta_1 = 45^\circ$ ($t = 0.0s$) conditions appeared to be the best in the pilot study. In the $10^\circ$ condition, the distraction rating was low, and frame rates were almost as fast as in the uniformly low detail condition. The $\theta_1 = 45^\circ$ condition had more participants and still had a low distraction rating. Also, no participants reported noticing the changing level of detail while performing the task in that condition. Frame rates were improved and search times were faster. There may be a trade off between faster frame rates and distraction caused by visible changing level of detail that caused both the most aggressive $\theta_1 = 10^\circ$ condition and the least aggressive $\theta_1 = 45^\circ$ condition to be rated the best.

The experimental design of the pilot study had a few limitations. Less control went into the design of the experiment. Participants were asked to perform only five searches in each condition. The number of participants in each condition was not held constant. Not every possible combination of independent variables was tested. Finally, because all participants saw the conditions in the same order, the results might have been affected by increasing comfort and experience with the virtual environment or search task. Participants will perform the search task in each condition twenty times instead of five in the full study. Participants will also experience the conditions in a random order. There will also be practice trials in the full study to reduce learning effects. Table 5 summarizes all of the changes to the experimental protocol of the full study as well as their motivation from the pilot study.
Table 5. Changes to the Experimental Protocol of the Full Study

<table>
<thead>
<tr>
<th>Change</th>
<th>Motivation</th>
</tr>
</thead>
<tbody>
<tr>
<td>No timeslots scheduled during mealtimes</td>
<td>Reduce the risk of simulator sickness</td>
</tr>
<tr>
<td>Eliminate participants who reported having issues “sometimes” or more often on three of the visual acuity conditions</td>
<td>Reduce the risk of simulator sickness</td>
</tr>
<tr>
<td>Larger object sizes</td>
<td>Search times were long and detail was hard to see</td>
</tr>
<tr>
<td>Fewer objects in the scene</td>
<td>Search times were long</td>
</tr>
<tr>
<td>Distribute the objects higher in the environment</td>
<td>Detail was hard to see in objects on the floor</td>
</tr>
<tr>
<td>More objects in the virtual environment</td>
<td>Frame rates in the pilot study were still fast enough to support interactivity even in the uniformly high detail condition</td>
</tr>
<tr>
<td>Test only eccentricity-based conditions</td>
<td>Attentional hysteresis did not improve task performance or reduce distraction.</td>
</tr>
<tr>
<td>Have participants perform more searches under each condition</td>
<td>More data will help establish more reliable conclusions</td>
</tr>
<tr>
<td>Randomize the order in which participants see the conditions</td>
<td>Results will not be subject to learning effects</td>
</tr>
<tr>
<td>Give participants a practice trial</td>
<td>Allow participant to acclimate to the environment before collecting data</td>
</tr>
</tbody>
</table>
CHAPTER 4. FULL STUDY

The purpose of the full study was to test the two most promising conditions in the pilot study more rigorously. Two conditions were chosen because both the most aggressive technique and the least aggressive technique in the pilot study showed promise. The design of the full study was more controlled than the design of the pilot study.

The full study focused more on the quantitative measures rather than the qualitative measures, although both were still collected. Participants performed a visual search task. Perceptual responses about the visual quality of the environment were collected. Search times, frame rates, and head positions for each frame were collected. Errors in detection were also counted. It was expected that perceptually adaptive rendering techniques would result in an improvement in quality over the uniformly high detail and uniformly low detail conditions.

4.1. Methods

4.1.1. Participants

Participants in the full study were drawn from the Summer 2007 Psychology department subject pool at Iowa State University and from fliers posted in Howe Hall. There were a total of 33 participants. Fifteen of the participants were female and eighteen of the participants were male. All had normal or corrected to normal vision and reported not being prone to motion sickness. Twenty-nine participants from the psychology pool received extra credit in a psychology course. Two participants received $10 for their participation in the experiment. The remaining two participants declined compensation.

Data from three of the participants was discarded. One participant did not feel comfortable with the virtual environment. A second participant’s data was discarded due to data collection errors. The third participant reported issues “sometimes” or higher on three visual acuity components. This participant was not allowed to enter the virtual environment
because they may have been at a higher risk for simulator sickness. The data presented in this chapter is from the remaining 30 participants.

4.1.2. Task

In the full study, participants were also required to search for a target object in a virtual living room environment. A virtual environment was constructed for this experiment using approximately 9,200,000 polygons (see Figure 52). The living room contained 20 objects, see Appendix B. There were four possible target objects: a monkey, a dragon, a snail, and a dog. The target objects are shown in Figure 50 and Figure 51. Some of the objects in the environment were duplicated, but only one target object was presented at any one time. The color of each of the 20 objects was randomly chosen from six colors: red, green, blue, cyan, magenta, and yellow. Randomly assigning object colors insured that the objects would require analysis of their shape, and not their color. This is important because shape is what is being manipulated. If the color were to remain constant, participants could easily find the object based on the color and not on the shape. Shape manipulations would be less likely to be noticed. When the target object was found, participants pressed a button on the game pad to stop a timer and reported the color of the object verbally to verify that the object had been found.
a) Dog Model  b) Dragon Model

Figure 50. Full Study Dog and Dragon Target Stimuli
a) Monkey Model  

b) Snail Model

Figure 51. Full Study Monkey and Snail Target Stimuli
4.1.3. Procedure

The procedure in the full study was the same as the procedure in the pilot study, described in section 3.1.3, with a few exceptions. Participants performed the search task in a practice condition first. This allowed the participants to become acclimatized to the virtual environment and the search task. Participants experienced three experimental conditions in the full study instead of the six conditions of the pilot study. The full study had twenty searches in each condition instead of the five searches of the pilot study. Figure 52 shows the virtual environment and objects used in the full study. Participants were taken out of the virtual environment after performing the search task in each condition in order to complete questionnaires.

Participants completed simulator sickness questionnaires at the beginning of the study and the end of each condition instead of only at the beginning and end of the study. The post-task questionnaires at the end of each condition were written instead of verbal. There was no end-of-the-study questionnaire.
Figure 52. Virtual Environment and Objects Used in the Full Study
4.1.4. Experimental Design

There were three different experimental conditions in the full study. One condition presented the environment in uniformly high detail. A second condition presented the environment in uniformly low detail. The third condition presented the environment with eccentricity-based perceptually adaptive rendering. The $\theta_1$ used in the perceptually adaptive rendering condition was either $45^\circ$ or $10^\circ$. The first twenty participants were in part 1 of the full study employing the $\theta_1 = 45^\circ$ perceptually adaptive condition. The last ten participants were in part 2 of the full study employing the $\theta_1 = 10^\circ$ perceptually adaptive condition. Attentional hysteresis was not used in the full study.

The target objects in the full study were a monkey, a snail, a dragon, and a dog. The monkey was the target object in the practice condition for all participants. For the three experimental conditions, the target object was randomly chosen. Participants searched for each target object once, but the condition associated with each target was not the same for each participant. The order in which participants experienced each condition was also randomized to avoid carryover effects, e.g. learning effects, from affecting one condition more than another.
4.2. Results

4.2.1. Visual Acuity

One participant reported issues with three of the visual acuity components “sometimes”. This participant was not allowed to complete the experiment in order to avoid the possibility of simulator sickness. Figure 53 shows the responses from the remaining participants on the visual acuity questionnaire. The majority of responses fell in the range where participants “rarely” had issues with visual acuity. Glare disability and light/dark adaptation had higher responses corresponding to “sometimes” having problems. Participants should not have experienced glare in the virtual environment. They were also not required to switch back and forth between a light and a dark environment. Visual search also received higher responses. This could be a cause of variation in participant search times and search distances. Overall, the responses in Figure 53 were very similar to the responses in the pilot study shown in Figure 42.
Figure 53. Visual Acuity Responses from the Full Study
4.2.2. Simulator Sickness

Participants in the full study exhibited higher symptoms of simulator sickness than the participants in Kennedy et al. (1993). For a discussion on why this may not be a reason for concern, see section 3.2.2. Participants in this study were novices while participants in Kennedy et al. were trained fighter pilots. The participants in this study also exhibited some symptoms of simulator sickness before entering the virtual environment. Average simulator sickness scores were lower in the full study than in the pilot study.

Figure 54 shows the average participant responses to the simulator sickness questionnaires for the different conditions in part 1. There was no main effect of display technique with regards to nausea \( F(1,19) = 0.17, MSE = 9.10, p = 0.68 \), oculomotor impairment \( F(1,19) = 0.03, MSE = 1.44, p = 0.87 \), disorientation \( F(1,19) = 0.04, MSE = 4.84, p = 0.85 \), or total sickness \( F(1,19) = 0.08, MSE = 5.60, p = 0.78 \) in part 1.

Figure 55 shows the average participant responses to the simulator sickness questionnaire for the different conditions in part 2. Participants in part 2 experienced less simulator sickness than participants in part 1. This is likely just due to individual differences between participants. There was no main effect of display technique with regards to nausea \( F(1, 19) = 1.71, MSE = 72.81, p = 0.22 \), oculomotor impairment \( F(1,19) = 0.10, MSE = 2.87, p = 0.76 \), disorientation \( F(1,19) = 0.47, MSE = 38.75, p = 0.51 \), or total sickness \( F(1,19) = 1.69, MSE = 34.27, p = 0.23 \) in part 2.

The average disorientation score was slightly but not significantly higher than the findings in Kennedy et al. Feelings of disorientation are not surprising for participants unfamiliar with virtual reality. The low detail condition appears to cause the most feelings of disorientation. The low detail condition required participants to concentrate more on the search in order to find the target object. This could result in participants feeling more tired or confused after that condition.
Figure 54. Comparison of Average Simulator Sickness Scores in Part 1

Figure 55. Comparison of Average Simulator Sickness Scores in Part 2
4.2.3. Qualitative Measures

Participants were asked if they noticed anything unusual about the visual quality of the virtual environment after they completed each round of search tasks. Forty percent of the participants noticed the changing level of detail in the $\theta_1 = 10^\circ$ condition. Forty-five percent of the participants noticed the changing level of detail in the $\theta_1 = 45^\circ$ condition. Participants exhibited frustration in the uniformly high detail condition because the game pad frequently did not respond to their button presses when they had found an object. It was observed that participants appeared confused in the low detail condition even if they could not express exactly why they were having difficulties. They knew that something was wrong that caused them to have difficulty finding the object, but were not able to identify the problem.

4.2.4. Part 1 Quantitative Results

The full study had two parts. In part 1, the $\theta_1 = 45^\circ$ perceptually adaptive condition was tested. In part 2, the $\theta_1 = 10^\circ$ perceptually adaptive condition was tested. Frame rate, distraction ratings, and number of errors were collected for each condition. Search times and distances were recorded for each trial that participants performed in each condition. Trials in which the color of the object was incorrectly identified were discarded. The data was analyzed using a repeated-measures analysis of variance. All post-hoc comparisons were adjusted using the Bonferroni correction for multiple comparisons.

Part 1 tested the less aggressive $\theta_1 = 45^\circ$ condition. Figure 56 shows the average search times, frame rates and distraction ratings. The average search distances and number of search errors are shown in Figure 57.

There was no main effect of display technique on search time ($F(1, 342) = 0.58, MSE = 23.11, p = 0.45$). The average search time for the uniformly low detail condition is slightly but not significantly higher than the average search times for the uniformly high detail and the perceptually adaptive conditions.
When frame rates were compared, a main effect of display technique was found ($F(1, 19) = 132.36, MSE = 5021.72, p < 0.001$). The mean frame rate for the low detail condition was significantly higher ($p < 0.001$) than the mean frame rate for the high detail condition. The mean frame rate for the high detail condition ($\mu = 3.43, SE = 0.19$) was significantly lower ($p < 0.001$) than the mean of the perceptually adaptive condition ($\mu = 25.84, SE = 1.85$). The mean of the perceptually adaptive rendering condition was also significantly lower than the low detail condition ($\mu = 50.93, SE = 0.81$).

There was no main effect of display technique on distraction ratings ($F(1, 18) = 0.56, MSE = 0.42, p = 0.47$). This is good because participants would complain if the adaptive technique was overly distracting.

A main effect of display technique was found for search distance ($F(1, 339) = 7.64, MSE = 335539.38, p = 0.006$). The average search distance in the perceptually adaptive condition ($\mu = 269.82, SE = 13.41$) was significantly shorter ($p = 0.029$) than the search distance for the low detail condition ($\mu = 335.60, SE = 23.20$). There was also a significant difference between the search distances in the perceptually adaptive condition and the high detail condition ($\mu = 225.39, SE = 10.43$). The perceptually adaptive condition had significantly longer search distances ($p = 0.018$) than the high detail condition.

There was no main effect of display technique on object detection errors ($F = 2.92, MSE = 3.60, p = 0.10$). This means that participants did not report the wrong color or make false button presses significantly more in any of the conditions. However, participants made on average approximately one less error in the perceptually adaptive condition than they did in the uniformly low condition.
Figure 56. Search Time, Frame Rates, and Distraction Ratings in Part 1 of the Full Study

(A red arrow indicates a significant post-hoc comparison. An asterisk indicates a significant main effect.)
Figure 57. Search Distance and Number of Search Errors in Part 1 of the Full Study

(A red arrow indicates a significant post-hoc comparison. An asterisk indicates a significant main effect.)
4.2.5. Part 2 Quantitative Results

Part 2 of the full study tested the more aggressive $\theta_1 = 10^\circ$ condition. Figure 58 shows the average search times, frame rates, and distraction ratings. Average search distances and number of search errors are shown in Figure 59.

There was a main effect of display technique on search times ($F(1,144) = 5.63, MSE = 110.08, p = 0.02$). The mean search time for the perceptually adaptive condition ($\mu = 7.13, SE = 0.40$) was significantly faster ($p = 0.009$) than the mean search time of the low detail condition ($\mu = 9.05, SE = 0.48$). There was a marginally significant difference ($p = 0.057$) between the perceptually adaptive rendering condition and the high detail condition ($\mu = 8.36, SE = 0.37$). Average search time in the $\theta_1 = 10^\circ$ perceptually adaptive condition was approximately one second faster than the average search time in the high detail condition.

There was a main effect of display technique on frame rates ($F(1, 9) = 4567.69, MSE = 8988.02, p < 0.001$). The mean frame rate of the perceptually adaptive rendering condition ($\mu = 46.37, SE = 0.65$) was significantly higher ($p < 0.001$) than the high detail condition ($\mu = 3.97, SE = 0.10$). There was also a significant difference ($p = 0.014$) between the perceptually adaptive condition and the low detail condition ($\mu = 51.32, SE = 1.08$). Frame rates in the perceptually adaptive condition were almost as high as the frame rates in the low detail condition.

No main effect of display technique was found with regards to distraction ratings ($F(1, 9) = 2.25, MSE = 0.20, p = 0.168$). It is good that there is no difference in distraction ratings in the three conditions. Participants would complain if the perceptually adaptive technique overly distracted them from performing the task.

There was no main effect of display technique on search distances ($F(1, 186) = 0.03, MSE = 544.07, p = 0.87$). However, search distances were numerically shorter in the perceptually adaptive condition than in the low detail condition.
There was no main effect of display technique on object detection errors ($F(1, 9) = 0.14, MSE = 0.20, p = 0.72$).
Figure 58. Search Times, Frame Rates, and Distraction Ratings in Part 2 of the Full Study

(A red arrow indicates a significant post-hoc comparison. An asterisk indicates a significant main effect.)
Figure 59. Search Distances and Number of Search Errors in Part 2 of the Full Study

(A red arrow indicates a significant post-hoc comparison. An asterisk indicates a significant main effect.)
4.3. Discussion

The purpose of the full study was to test perceptually adaptive techniques more rigorously in an experiment with more control built into the design. A less aggressive $\theta_1$ of $45^\circ$ and a more aggressive $\theta_1$ of $10^\circ$ were chosen for testing. Both conditions were studied because they both performed well in the pilot study.

The study was broken up into two parts. In the first part, the $45^\circ$ condition was tested. In the second part, the $10^\circ$ condition was tested. Participants performed a visual search task in each of the three experimental conditions, a uniformly high detail condition, a uniformly low detail condition and an adaptive condition. The amount of computational resources available for each condition was the same providing a source of experimental control. The order in which participants experienced the conditions was randomized so that the results would not be subject to learning effects.

There was no significant difference in distraction ratings among the conditions in either part 1 or part 2 of the full study. There may have been a small, but non-significant increase in distraction in the $\theta_1=45^\circ$ perceptually adaptive condition. This means that employing perceptually adaptive techniques did not distract the participants. If these techniques had been distracting, then participants would have complained.

The average angular distance covered by participants while searching for objects was significantly different in the $\theta_1 = 45^\circ$ condition, participants moved their head 50 degrees more in the perceptually adaptive condition than in the high detail condition. In the $\theta_1 = 10^\circ$ condition, there was no significant difference in search distance. This is most likely because in the $45^\circ$ condition, the switch from high to medium detail occurs in the viewer’s peripheral vision. The human eye is especially sensitive to motion in the periphery (Longhurst, Debattista, & Chalmers, 2006). The sensation of motion which can occur when levels of detail change may have attracted the user’s attention and made them turn their head more in the $45^\circ$ condition. In the $10^\circ$ condition, the level of detail switch was more towards the
center of our field of view and thus less likely to cause participants to move their head to look at the changing object in the periphery. However, search distances in both of the perceptually adaptive conditions were much less than in the uniformly low detail condition.

Search times in the perceptually adaptive conditions were equal to if not faster than search times in the high detail condition. Average times were fastest in the 10° perceptually adaptive condition. This is because frame rates were faster, and detail was still being shown where it was needed (i.e. where the participant was looking). Participants did not need to wait for the view to update from their previous head movement in order to make their next head movement. Also, the faster update rates meant that the region of higher detail was better able to follow head movements.

Frame rates of the perceptually adaptive conditions in the full study were significantly faster than the frame rates in the uniformly high detail condition. Frame rates in the high detail condition were well below levels of acceptable interactivity. Participants were very frustrated in the high detail condition when they had to repeatedly press the button because the button presses were not registered. Because of the very low frame rates, it took approximately 276 ms for the action of the user to be reflected in the environment. Frame rates in the \( \theta_1 = 10° \) condition were almost equal to the uniformly low detail frame rates. In this condition, it only took 46 ms for the environment to react to actions in the environment.

While the \( \theta_1 = 45° \) perceptually adaptive rendering condition did provide shorter and therefore better search distances than the low detail condition, the search distances were not as short as the high detail condition. Search times in the perceptually adaptive condition were numerically faster, but not enough to be significant.

The \( \theta_1 = 45° \) perceptually adaptive rendering condition in part 1 showed faster frame rates than the high detail condition as expected. Search times were not significantly different between the two conditions. This means that search task was not impeded by using perceptually adaptive techniques. The increase in frame rate caused by the implementation
of perceptually adaptive rendering makes the $45^\circ$ eccentricity threshold a viable solution for improving interactivity.

The $\theta_1 = 10^\circ$ perceptually adaptive rendering condition in part 2 showed more even improvement over the high detail condition. Frame rates were almost as high as those in the low detail condition. Search times were also one second faster. There was no difference in search distance covered. This indicates that this perceptually adaptive condition is the best for use in immersive virtual reality applications.
CHAPTER 5. CONCLUSION

5.1. Summary

This research was aimed at designing and validating a perceptually adaptive rendering system for immersive virtual reality environments. The rendering system allows for rapid environment authoring. Environments can be created, saved and replayed without recompiling the source code. Three discrete levels of detail are used. The appropriate level of detail is selected using an eccentricity-based technique. The eccentricity-based technique can be augmented with an attentional hysteresis factor.

The perceptually adaptive rendering system was validated through a pilot study and a full study. The aim of the pilot study was to uncover trends in performance and perceptual quality when the eccentricity factor and the attentional hysteresis factor were manipulated. The pilot study showed that both the $\theta_1 = 10^\circ$ and the $\theta_1 = 45^\circ$ degree conditions provided benefit. The attentional hysteresis factors tested did not improve search times or frame rates.

The full study tested both the $\theta_1 = 10^\circ$ and the $\theta_1 = 45^\circ$ degree conditions more rigorously. It was shown that both the $45^\circ$ and the $10^\circ$ conditions are able to improve the quality of the environment. Frame rates were significantly faster in the perceptually adaptive conditions than in the high detail condition. Frame rates were improved by over ten times in the $10^\circ$ condition as compared to the high detail condition. Search times in the perceptually adaptive conditions were also equal to if not faster than search times in the high detail condition. There was no significant difference in distraction ratings. These results indicate that perceptually adaptive rendering is a useful technique for improving both system and user performance in immersive virtual reality environments.

If the virtual environment requires user interaction, then perceptually adaptive rendering techniques should be used. The $\theta_1 = 10^\circ$ would be best for environments designed for one user because of the smaller region of high detail. The $\theta_1 = 45^\circ$ condition, which still
allows for interactive frame rates, would be more useful when multiple people are viewing the virtual environment together.

It is expected that perceptually adaptive rendering systems will become more prevalent. Three-dimensional scanning technologies are becoming more commonplace and allow for the capture of high detail representations of real objects. In order to employ these high detail models, perceptually adaptive rendering can be used to generate a high quality environment while not impairing system or task performance. It would be beneficial to apply perceptually adaptive rendering techniques to visualization of large data sets. The area of data that is attended would be displayed in high detail so that the actual data can be seen. In the periphery the data display could be simplified. This will provide enough information to uncover trends, but the rendering hardware will not be burdened. Perceptually adaptive rendering techniques could also be applied in surgery aided by virtual reality. Only a small area of the body on which the surgeon is currently operating would need to be rendered in high detail. The detail in the rest of the scene could be reduced without impairing the surgeon’s efficacy.

The two studies used to validate the perceptually adaptive rendering system described in this thesis were based on a visual search task. It is assumed that testing the rendering system with a different task would result in the same benefits. Search tasks are often used when evaluating perceptually adaptive rendering systems. Also a search task is a rigorous test of perceptually adaptive rendering techniques because it requires the user to focus on the detail of the objects in order to correctly detect the target object. If the detail is not there, it is hard to determine whether or not a given object is the target.

The virtual environment constructed for the full study was so polygon-intensive, that frame rates were severely impacted in the uniformly high detail condition. This was done in order to create the most dramatic improvements in the perceptually adaptive condition. This does not make the results of the study less valid. The low frame rates in the uniformly high
detail condition actually serve as an indicator of how much environments displayed on older systems can be improved by using perceptually adaptive techniques.

Most of the models used in the two validation studies had relatively smooth features. This is because the model scanner employed was unable to adequately capture objects with extreme surface changes. Therefore there is little perceptual difference between the high detail and the medium detail versions of the models even though the high detail version of a model had one hundred times more triangles than the medium detail version of the same model. The dragon and the happy Buddha models were acquired from an online database of high detail models and thus had more surface variability. Since participants also viewed these models and were not impacted by the larger surface variation, it is assumed that the evaluation of perceptually adaptive rendering techniques in this thesis is valid.

**5.2. Benefit in Older and Newer Systems**

It is clear that perceptually adaptive rendering techniques can be beneficial for older hardware systems. In older systems, the goal of perceptually-driven simplification methods would be to reduce the time necessary to render a single frame in order to improve interactivity for polygon intensive environments. Environments which would otherwise be impossible to display on older hardware could be displayed if perceptually adaptive rendering techniques are used. Fewer vertices would need to be rendered and thus frame rates would be raised. These improved frame rates can also lead to a reduction in simulator sickness. This is because actions taken by the user will be reflected more quickly by the virtual environment if the computers are unburdened.

The graphics cards in newer systems are able to feed data through the pipeline at a much higher rate, so it may seem that level of detail simplification methods are becoming less vital. If more data can be processed faster, then low frame rates may not be a problem in new systems. Benefit can still be derived from using spared resources to give even more
quality to the portions of the scene that are being attended. Also, reducing the computational demand can save electricity, making level of detail simplification a cost effective strategy which would be beneficial for older and newer systems (Reingold, Loschky, McConkie, & Stampe, 2003).

5.3. **Clustered Systems**

An unsolved issue with the use of perceptually-driven simplification on fully immersive virtual environments is how to take advantage of unburdened resources when using clustered systems. Even though it is unnecessary to render high detail for the portion of the scene that is not being attended (especially the part of the scene which is behind the user’s head), reducing the complexity will only reduce the bottlenecks for some of the computers. The computers responsible for the portion of the scene that is being attended will still tend to be burdened. In order to receive the most benefit from level of detail simplification, there needs to be a way of using the unburdened resources on some computers to improve image quality for the attended portions of the environment rendered by other computers. This means that all frames have to be locked so that no computer in the cluster renders a new image until all of the computers in the cluster are ready to display that frame. Otherwise, the image quality might deteriorate instead. This is because the images on each wall will not match up with the images on adjacent walls. More research needs to be carried out to find ways of shifting computational resources between the various computers that make up a clustered system.

One area where the unburdened computers could be used is lighting calculations. If the computers responsible for the attended portion of the scene are being overburdened, the freed resources on one of the other computers could be used to create a lighting map. This map could then be sent to the burdened computer. Instead of performing the lighting
calculations on each vertex, the burdened computer would employ the calculations provided by another computer.

It could also be possible to use an unburdened computer to perform the physics calculations for attended portion of the environment. The graphics processing unit (GPU) on a computer can perform certain kinds of calculations much faster than the computer’s central processing unit (CPU). GPUs are also capable of performing physics calculations through the use of special vertex and fragment programs (Macedonia, 2006). The unburdened GPU on a computer responsible for rendering unattended portions of the scene could utilize these programs to calculate physics for the attended portion of the scene. The results of these calculations could then be sent to the burdened computer and applied.

The unburdened processor could also be used to help with predictive level of detail selection for the next frame. The freed resources can be occupied with computing saliency maps for the next frame which could be sent to all of the computers in the clustered system for use in determining which level of detail to employ. The saliency maps can augment a reactive level of detail scheme. Objects near the current focus of attention that are likely to attract attention would be rendered at a higher detail to reduce popping affects when focus is changed one of those objects. It might even be possible to replace the reactive level of detail scheme eliminating the need to receive information from the head or eye tracker before proceeding with computation.

5.4. Future Directions

The goal of this research was to design and evaluate a perceptually adaptive rendering system that works in an immersive virtual reality environment. It was shown that an eccentricity-based selection strategy using discrete geometric level of detail improves frame rates. This technique does not impede and in some cases reduces search times and distances
without overly distracting the user. There are still areas where the rendering system could be extended.

While there has been much research in recent years on the effects of geometric and simulation simplification on task performance, the area of shader simplification is still completely unexplored. More work needs to be put into investigating the difference in cost of switching between shader levels of detail and using the same more complicated shader for every situation. Also, distinctive patterns are features that draw one’s attention to an object. Research needs to be performed to determine if reduction of material complexity will negatively affect search times.

More benefit could be derived from using continuous level of detail instead of the discrete level of detail method employed in this research. In continuous level of detail methods, adjustments to the shape of the object are made constantly as a function of angular distance instead of just when the angular distance exceeds certain threshold angles. This causes the shape to change gradually instead of all at once the way it does using discrete methods. Continuous level of detail was not employed in this research because software could not be found which was compatible with the software used to run the application in immersive virtual reality and updated the vertices fast enough for real-time rendering.

One of the advantages of fully immersive virtual reality environments is that multiple users can view the same environment together. It would be beneficial to test the perceptually adaptive rendering system with multiple users simultaneously viewing the environment. There are two ways the rendering system could be adjusted to accommodate multiple users. In one scenario, larger theta values could be used so that more of the virtual environment is preserved at higher levels of detail. This might be beneficial when the person wearing the head tracking device, the driver, is leading other viewers in an environment. In this situation, the driver acts as a tour guide direction the attention of the others. A larger theta value may be sufficient in this situation. In another scenario, multiple users could have their head
tracked so that all of the viewers of the scene see high detail in the direction they are looking. In this method, a smaller theta value could be used. This method would be better for scenarios when the viewers in the environment are working independently. It would be interesting to see how many users, head-tracked or not, can view the perceptually adaptive virtual environment while maintaining acceptable frame rates and without revealing large areas of low detail to them.

Attentional hysteresis was dismissed after the pilot study because frame rates suffered and search times were not improved. This may have been the case because the delay values tested were too large. Also, the search task employed may not have been the best test of this method. In the visual search task used in the pilot study, participants were asked to find the target object as quickly and accurately as possible. This means that once a given object was rejected as a candidate for the target object, it was unlikely to be revisited until the other objects in the scene had first been rejected. Attentional hysteresis on the other hand is based upon the fact that objects that were once the object of attention are likely to be the object of attention again. Attentional hysteresis may be better evaluated using a different kind of task that requires the user to analyze the environment more thoroughly.
APPENDIX A. SHADER LICENSE

Copyright (C) 2002-2005 3Dlabs Inc. Ltd.
All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

Neither the name of 3Dlabs Inc. Ltd. nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDERS OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
APPENDIX B. MODELS USED IN THE PILOT AND FULL STUDIES

Car

Cat

Dog

Dragon

Duck
APPENDIX C. VISUAL ACUITY QUESTIONNAIRE

1. I have problems adjusting to bright room lighting, after the room lighting has been rather dim.
   \[\begin{array}{ccccc}
   & 1 & 2 & 3 & 4 & 5 \\
   Never & Rarely & Sometimes & Often & Always \\
   \end{array}\]

2. I have trouble noticing things in my peripheral vision.
   \[\begin{array}{ccccc}
   & 1 & 2 & 3 & 4 & 5 \\
   Never & Rarely & Sometimes & Often & Always \\
   \end{array}\]

3. I have trouble finding a specific item on a crowded supermarket shelf.
   \[\begin{array}{ccccc}
   & 1 & 2 & 3 & 4 & 5 \\
   Never & Rarely & Sometimes & Often & Always \\
   \end{array}\]

4. I have problems with lights around me causing glare when I’m trying to see something.
   \[\begin{array}{ccccc}
   & 1 & 2 & 3 & 4 & 5 \\
   Never & Rarely & Sometimes & Often & Always \\
   \end{array}\]

5. I tend to confuse colors.
   \[\begin{array}{ccccc}
   & 1 & 2 & 3 & 4 & 5 \\
   Never & Rarely & Sometimes & Often & Always \\
   \end{array}\]

6. I have trouble locating a sign when it is surrounded by a lot of other signs.
   \[\begin{array}{ccccc}
   & 1 & 2 & 3 & 4 & 5 \\
   Never & Rarely & Sometimes & Often & Always \\
   \end{array}\]

7. I have problems reading small print (for example, phone book, or newspapers).
   \[\begin{array}{ccccc}
   & 1 & 2 & 3 & 4 & 5 \\
   Never & Rarely & Sometimes & Often & Always \\
   \end{array}\]

8. I have trouble reading a sign or recognizing a picture when it’s moving, such as an ad on a passing bus or truck
   \[\begin{array}{ccccc}
   & 1 & 2 & 3 & 4 & 5 \\
   Never & Rarely & Sometimes & Often & Always \\
   \end{array}\]

9. When pouring a liquid, I have trouble judging the level of the liquid in a container, such as the level of coffee in a cup.
   \[\begin{array}{ccccc}
   & 1 & 2 & 3 & 4 & 5 \\
   Never & Rarely & Sometimes & Often & Always \\
   \end{array}\]

10. I have trouble reading the menu in a dimly lit restaurant.
    \[\begin{array}{ccccc}
    & 1 & 2 & 3 & 4 & 5 \\
    Never & Rarely & Sometimes & Often & Always \\
    \end{array}\]
11. I have trouble seeing moving objects coming from the side until they are right in front of me.

<table>
<thead>
<tr>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>Never</td>
<td>Rarely</td>
<td>Sometimes</td>
<td>Often</td>
<td>Always</td>
</tr>
</tbody>
</table>

12. It takes me a long time to adjust to darkness after being in bright light.

<table>
<thead>
<tr>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>Never</td>
<td>Rarely</td>
<td>Sometimes</td>
<td>Often</td>
<td>Always</td>
</tr>
</tbody>
</table>

13. When I’m driving, other cars surprise me from the side, because I don’t notice them until the last moment.

<table>
<thead>
<tr>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>DO NOT DRIVE</td>
<td>Never</td>
<td>Rarely</td>
<td>Sometimes</td>
<td>Often</td>
<td>Always</td>
</tr>
</tbody>
</table>

14. I have trouble driving when there are headlights from oncoming cars in my field of view.

<table>
<thead>
<tr>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>DO NOT DRIVE</td>
<td>Never</td>
<td>Rarely</td>
<td>Sometimes</td>
<td>Often</td>
<td>Always</td>
</tr>
</tbody>
</table>

15. I have difficulty reading small print under poor lighting.

<table>
<thead>
<tr>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>Never</td>
<td>Rarely</td>
<td>Sometimes</td>
<td>Often</td>
<td>Always</td>
</tr>
</tbody>
</table>

16. I have problems locating something when it’s surrounded by a lot of other things.

<table>
<thead>
<tr>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>Never</td>
<td>Rarely</td>
<td>Sometimes</td>
<td>Often</td>
<td>Always</td>
</tr>
</tbody>
</table>

17. The color names that I use disagree with those that other people use.

<table>
<thead>
<tr>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>Never</td>
<td>Rarely</td>
<td>Sometimes</td>
<td>Often</td>
<td>Always</td>
</tr>
</tbody>
</table>

18. I have problems carrying out activities that require a lot of visual concentration and attention.

<table>
<thead>
<tr>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>Never</td>
<td>Rarely</td>
<td>Sometimes</td>
<td>Often</td>
<td>Always</td>
</tr>
</tbody>
</table>

19. When I’m walking along, I have trouble noticing objects off to the side.

<table>
<thead>
<tr>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>Never</td>
<td>Rarely</td>
<td>Sometimes</td>
<td>Often</td>
<td>Always</td>
</tr>
</tbody>
</table>

20. It takes me a long time to find an item in an unfamiliar store.

<table>
<thead>
<tr>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>Never</td>
<td>Rarely</td>
<td>Sometimes</td>
<td>Often</td>
<td>Always</td>
</tr>
</tbody>
</table>
21. Sometimes when I reach for an object, I find that it is further away (or closer) than I thought.
   1   2   3   4   5
   Never  Rarely  Sometimes  Often  Always

22. I have difficulty noticing when the car in front of me is speeding up or slowing down.
   1   2   3   4   5
   Never  Rarely  Sometimes  Often  Always

23. It takes me a long time to adjust to bright sunshine after I have been inside a building for a lengthy period of time.
   1   2   3   4   5
   Never  Rarely  Sometimes  Often  Always

24. When driving at night, objects from the side unexpectedly appear or pop up in my field of view.
   0   1   2   3   4   5
   DO NOT DRIVE Never  Rarely  Sometimes  Often  Always

25. I have difficulty distinguishing between colors.
   1   2   3   4   5
   Never  Rarely  Sometimes  Often  Always

26. I bump into people in a busy store because I have problems seeing them in my peripheral vision.
   1   2   3   4   5
   Never  Rarely  Sometimes  Often  Always

27. I have difficulty doing any type of work which requires me to see well up close.
   1   2   3   4   5
   Never  Rarely  Sometimes  Often  Always

28. I have trouble adjusting from bright to dim lighting, such as when going from daylight into a dark movie theater.
   1   2   3   4   5
   Never  Rarely  Sometimes  Often  Always

29. When driving at night in the rain, I have difficulty seeing the road because of headlights from oncoming cars.
   0   1   2   3   4   5
   DO NOT DRIVE Never  Rarely  Sometimes  Often  Always
30. When riding in a car, other cars on the road seem to be going too fast.

<table>
<thead>
<tr>
<th>Never</th>
<th>Rarely</th>
<th>Sometimes</th>
<th>Often</th>
<th>Always</th>
</tr>
</thead>
</table>

31. I find it difficult changing lanes in traffic because I have trouble seeing cars in the next lane.

<table>
<thead>
<tr>
<th>DO NOT DRIVE</th>
<th>Never</th>
<th>Rarely</th>
<th>Sometimes</th>
<th>Often</th>
<th>Always</th>
</tr>
</thead>
</table>

32. I have problems judging how close or far things are from me.

<table>
<thead>
<tr>
<th>Never</th>
<th>Rarely</th>
<th>Sometimes</th>
<th>Often</th>
<th>Always</th>
</tr>
</thead>
</table>

33. It takes me a long time to get acquainted with new surroundings.

<table>
<thead>
<tr>
<th>Never</th>
<th>Rarely</th>
<th>Sometimes</th>
<th>Often</th>
<th>Always</th>
</tr>
</thead>
</table>
APPENDIX D. SIMULATOR SICKNESS QUESTIONNAIRE

Please report the degree to which you experience each of the below symptoms as one of "None", "Slight", "Moderate" and "Severe". Using a scale from "0" (none) to "3" (severe).

<table>
<thead>
<tr>
<th>Symptom</th>
<th>None</th>
<th>Slight</th>
<th>Moderate</th>
<th>Severe</th>
</tr>
</thead>
<tbody>
<tr>
<td>General Discomfort</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>Fatigue</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>Headache</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>Eyestrain</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>Difficulty focusing</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>Increased salivation</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>Sweating</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>Nausea</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>Difficulty concentrating</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>Fullness of head</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>Blurred vision</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>Dizzy (eyes open)</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>Dizzy (eyes closed)</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>Vertigo</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>Stomach awareness</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>Burping</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
</tbody>
</table>
APPENDIX E. POST-TASK QUESTIONNAIRE

Circle the most appropriate response.

1. While performing this task did you notice anything strange about the visual quality of the environment?

2. How much did the visual display quality interfere or distract you from performing the assigned task?

   1  2  3  4  5
   Not much Moderately Very much
APPENDIX F. END OF STUDY QUESTIONNAIRE

Circle the number most appropriate.

1. How often do you play video games?
   1. Not often  2. Moderately often  3. Very often

2. How responsive was the environment to actions that you initiated (or performed)?

3. How aware were you of events occurring in the real world around you?

4. How much did your experiences in the virtual environment seem consistent with your real-world experiences?

5. How compelling was your sense of moving around inside the virtual environment?

6. How closely were you able to examine objects?

7. How well could you examine objects from multiple viewpoints?
   1. Not well  2. Moderately well  3. Very well

8. To what degree did you feel confused or disoriented at the beginning of breaks or at the end of the experimental session?

9. How involved were you in the virtual environment experience?
10. How distracting was the control mechanism?

1  2  3  4  5
Not distracting  Moderately distracting  Very distracting

11. How much delay did you experience between your actions and expected outcomes?

1  2  3  4  5
No delay  Moderate delay  A lot of delay

12. How quickly did you adjust to the virtual environment experience?

1  2  3  4  5
Not quickly  Moderately quickly  Very quickly

13. How much did the control device interfere with performance of assigned tasks or other activities?

1  2  3  4  5
Not much  Moderately  Very much

14. How well could you concentrate on the assigned tasks or required activities rather than on the mechanisms used to perform those tasks or activities?

1  2  3  4  5
Not well  Moderately well  Very well

15. Did you learn new techniques that enabled you to improve your performance?

1  2  3  4  5
Not much  Moderately  Very much

16. Were you involved in the experimental task to the extent that you lost track of time?

1  2  3  4  5
Not much  Moderately  Very much

17. In the box provided write what you think this study was about?
**BIBLIOGRAPHY**


Gamma, E., Helm, R., Johnson, R., & Vlissides, J. (2002). Design Patterns: Elements of Reusable Object-Oriented Software. Indianapolis: Addison-Wesley.


Displays Consortium, Advanced Displays and Interactive Displays Third Annual Symposium, (pp. 79-83). College Park, MD.


