COVER SHEET

This is a manuscript version of this paper. The paper was first published as:

Brown, Ross A. and Cooper, Luke and Pham, Binh L. (2003) Visual Attention-based Polygon Level of Detail Management. In *Proceedings GRAPHITE2003 - International Conference on Computer Graphics and Interactive Techniques in Australiasia and South East Asia*, pages pp. 55-62, Melbourne.

Copyright 2003 ACM Press

Accessed from **http://eprints.qut.edu.au**

Visual Attention-based Polygon Level of Detail Management

Ross Brown
Faculty of Information Technology
QUT
r.brown@qut.edu.au

Luke Cooper
Faculty of Information Technology
QUT
l.cooper@student.qut.edu.au

Binh Pham
Faculty of Information Technology
QUT
b.pham@qut.edu.au

Abstract

Modern real-time graphics systems are required to render millions of polygons to the screen per second. However, even with this high polygon rendering bandwidth, there are still applications which tax this rendering capability. We introduce in this paper a technique which adaptively allocates polygons to objects in a scene according to their visual importance. It is expected that using this technique, an improvement in the perceptual quality of a rendered image should result, for the same overall number of polygons being rendered.

We present both a theoretical basis and a complete design for a visual attention-based level of detail management technique. We also present some preliminary assessment of output from the system. Applications for this technique are expected to be found in the areas of entertainment, visualisation and simulation.

CR Categories: I.3.7 (Three-Dimensional Graphics and Realism): Animation, Virtual reality; I.3.3 (Picture/Image Generation) Display algorithms, Viewing algorithms.

Keywords: Visual Attention, Level of Detail Management, Real-time Graphics.

1. Introduction

Real-time animated computer graphics has a very heavy reliance on the state of the current generation of graphics hardware. However, like many fields in computing science, the requirements of computer graphics software far outstrips hardware capabilities. Software developers are forced to turn to other methods to extract peak performance out of their software systems.

In the field of computer graphics, the primary role of graphics hardware is to display as many triangles as possible on a display device [Watt 1992]. Graphics programmers use the triangles to model and display three-dimensional objects. Real-time graphics systems place a heavy burden on underlying hardware capabilities, often requiring that many millions of triangles are displayed every second. However, there is always going to be a limit to the number of triangles the hardware can display. Once this limit is reached, the quality of the animated sequences is compromised.

Since the mid nineteen-seventies, programmers have used *Level Of Detail (LOD)* management to improve the performance and quality of their graphics systems [Reddy 1999]. The LOD approach involves maintaining a set of representations of each polygonal object, each with varying levels of triangle resolution [Clark 1976]. During the execution of the animation, objects deemed to be less important are displayed with a low-resolution representation. Whereas objects of higher importance are displayed with higher levels of triangle resolution.

LOD management refers to the criteria by which objects are assigned importance. Traditionally, these criteria have been based solely on distance or projected screen space [Reddy 1999]. Both of these methods presume that as objects appear smaller on screen, viewers are less able to distinguish detail. Displaying selected objects at lower detail frees up triangles that can be used elsewhere, or saves milliseconds of execution time that help to improve performance.

In this paper we present an extension to present methods of polygon LOD management. The approach presented in this paper uses a measure of the visual importance of the object in question to modulate the LOD assigned to the object. We present a model of visual attention suitable for real-time rendering systems, and then apply this model to the task of controlling the polygon count of objects contained within the scene. We believe that the perceptual quality of the scene will improve for a static number of triangles, if the distribution of triangles in the scene is biased so that more triangles are in the regions attended by the viewer. A corollary of this is also the possibility of no difference being detected between biased and non-biased images, which is still useful for polygon decimation systems.

The structure of the paper is as follows. Section 2 details the previous relevant work in this area. Section 3 details the visual attention theory used to formulate a real-time visual attention model. Section 3 details how this model is incorporated into the LOD management system in Performer, a visualisation system produced by Silicon Graphics [Eckel 2001]. Section 4 presents some preliminary assessment of the implementation so far. Section 5 concludes the paper with a discussion of achievements and future work.

2. Previous Work

LOD approaches provide different representations of the same object, with each representation having a different level of complexity [Reddy 1999]. For the purpose of this paper, complexity will refer to the number of polygons or triangles that make up the whole object. The reason for providing different representations at various resolutions is to facilitate *LOD Management* techniques that adapt the rendering parameters to optimise a particular performance aspect of a real-time graphics system. In this case, we wish to improve the perceptual quality of the image being generated.

LOD techniques are the methods used to generate the multiple representations of polygonal objects. Two types of LOD techniques presently used are d*iscrete* and c*ontinuous LODs*. A proposal for the discrete multi-resolution representation of polygonal models was made as early as 1976 [Clark 1976]. This approach suggested the use of stored representations of an object that can be quickly interchanged at run-time. This practice has since become standard in computer graphics systems [Luebke 2001; Reddy 1995] (refer to Figure 1).

Continuous LODs aim to increase or decrease the resolution of a polygon mesh through a series of well-defined operations on primitive graphics types, such as vertices and edges [Hoppe 1996]. Examples of continuous level of detail techniques are progressive meshes and topology simplification. Progressive meshes are defined as a series of edge collapse and vertex split operations [Hoppe 1996]. Models can be increased in resolution through vertex splits and decreased in resolution through edge collapses. Topology simplification algorithms seek to reduce polygonal models in a structural manner as opposed to a geometric manner [El-Sana 1998]. They do this by gradually eliminating high frequency details [He 1996].

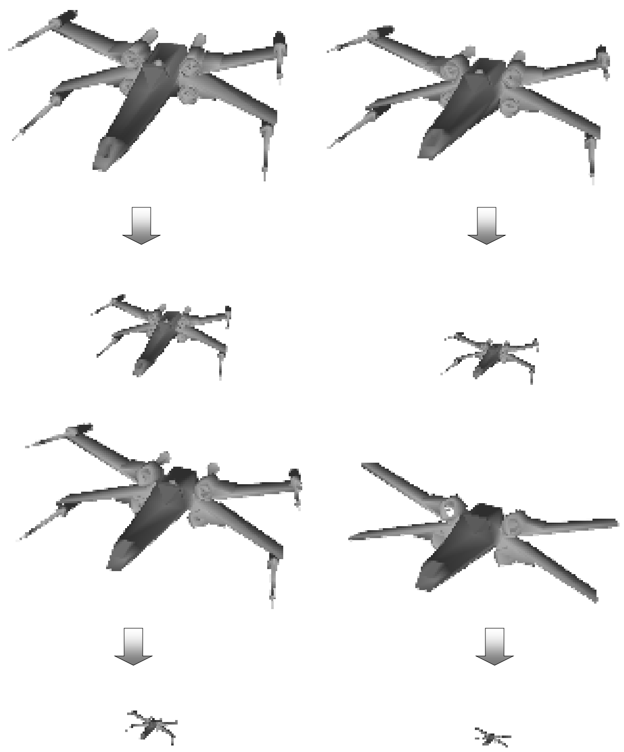

Figure 1 Four different representations of a spacecraft. The first and third rows shows the spacecraft full sized, while the second and fourth rows shows the final projected spacecraft. Note the similar perceived quality of the projected objects.

LOD management, therefore, is the method by which the detail levels are assigned to objects. Traditional LOD management techniques have used a simple but efficient selection method based around the concept of perceived object size [Reddy 1999; Reddy 1997]. Two of the most popular implementations of this concept are to use either a distance measure to the object [Reddy 1999] or the projected screen space of the object in question [Wernecke 1994]. Other techniques include using eccentricity and/or velocity [Oshima 1996; Funkhouser 1993], or enforcing an LOD level due to a required frame rate [Funkhouser 1993].

The first technique, using a distance measure, is most commonly used in applications where speed is of utmost importance, such as computer games [LaMothe 1999]. The LOD of an object is therefore a function of its distance from the viewport [Reddy 1999]. The reasoning being; the further away from the viewport an object is, the smaller it will appear to a viewer. Coarser representations can be used for far away objects, as the viewer will not notice any loss in quality. Objects that are closer will appear larger and so are rendered in finer detail [Clark 1976].

A similar method is to use the area of the projected screen space of the object. The screen space of an object is calculated by projecting its bounding box onto the viewport, then calculating the area of a screen-aligned rectangle that completely encloses the projected bounding box [Wernecke 1994]. The object representation is then chosen from this calculated area by comparing it to some precomputed threshold values.

Visual Attention and 3D Computer Graphics

The fields within computer graphics research that have made the most use of visual attention theory are in the areas of image/video processing and photorealistic image synthesis. Much of the work in image processing has been basic research aimed at the detection of regions of interest to model human visual attention [Osberger 2001; De Almeida Neves 2000; Milanese 1993; Itti 1998]. However, some have gone further and utilised the regions of interest to perform optimisations on image processing algorithms, usually in the areas of image and video compression [Osberger 2001]. Here, the compression is applied heavily to regions that are not regarded by viewers. Thus, for an image sequence of the same compressed size, the image with the compression rate modulated by a visual attention model appears subjectively to be of a higher quality than the image with spatially uniform compression.

There has also been research performed on optimising various image synthesis algorithms with visual attention-based approaches [Brown 1002; Yee 2001; Horvitz 1997]. One such algorithm included modulating the super-sampling performed by a ray-tracer [Brown 2001]. Subjective testing has shown that the distortions introduced into the image did not disturb the perceived quality of the image. Similar work has been performed with a Monte Carlo global illumination algorithm [Horvitz 1997]. However, the image quality of the resultant output was not subjectively tested. Both algorithm optimisations produced significant performance increases, and indicate that the perceptual quality of a scene is influenced by the appropriate application of rendering resources to regions being regarded by the viewer.

Work has been performed into real-time management of LODs via a visual attention model [Horvitz 1997], that extends previous work [Funkhouser 1993] attempting to control the frame rate of an animation by predicting the complexity of the frame to be rendered. The adapted model takes into consideration the perceptual degradation of various aspects of the scene due to visual attention focus. The work includes a brief discussion of level of detail degradation, but fails to explore this notion further, concentrating instead on 2D and 3D sprite manipulation [Horvitz 1997].

We note that the successful application of visual attention-biased methods to image synthesis/processing is evidence for a successful application of the same principles to LOD management. We now describe our own novel method of visual attention-biased LOD management for geometry meshes.

3. A New Level of Detail Management Technique Incorporating a Model of Visual Attention

This paper introduces a novel approach to level of detail management that goes beyond a simple size measurement. The proposed method seeks to predict viewer eye movements while regarding a generated scene. These eye movement predictions identify regions of interest on the screen. These regions of interest can then be mapped to polygonal objects within the scene and used to assign per-object importance levels. As mentioned previously, these importance levels are used to assign levels of detail.

The eye movement predictions are calculated using a model derived from theories of *visual attention*. Visual attention theory is a broad subject that attempts to explain the attentive behaviour of humans [Wolfe 1996]. That is, it explains how humans select the objects upon which they focus their attention. Low-level visual features often direct attention to spatial locations. Examples of these visual features are: movement, luminance contrast and size [Wolfe 1996]. These features can be easily extracted from three-dimensional scenes, and can then be used to assign per-object importance.

This particular LOD management technique has not been attempted before. Therefore, the main objective behind this research is to determine if the incorporation of visual attention into LOD management is successful. The hypothesis is that there will be a noticeable improvement in the subjective quality of an animated sequence. This is related to the fixed bandwidth available to the graphics card for rendering, where the technique is expected to work in a similar manner to the image compression methods listed in Section 2 [Osberger 2001]. That is, when the triangles are distributed in a spatially non-uniform manner to regions of visual importance, then the visual quality of the image will be improved. The system may also have application in seeking to improve the efficiency of rendering systems by removing polygons in areas not noticed by the viewer [Brown 2001].

The first step in testing this hypothesis is the design of a theoretical framework for LOD management that allows for modulation by an importance value assigned to the object. Secondly, a model of visual attention must be designed which is able to garner visual features from a scene database and then assign a visual importance value on an object by object basis. Finally, both of these have to be incorporated into a real-time triangle-based polygon rendering system.

The LOD management technique has been developed to work within a simulation software library developed by Silicon Graphics, called Performer [Eckel 2001]. Performer was chosen due to its built in hooks for modification of LOD management policies, and further API interfaces to its frame rate control systems. This has provided an ideal support for the development of this visual attention based LOD management technique. We now describe this technique in detail.

3.1. Level of Detail Management Framework

The following diagram shows the frame rate modes supported by Performer. The last mode, *PFPHASE_LOCK*, gives a good visual description of the system implemented by this project. The bold vertical lines indicate the set frame-rate time intervals, for example, 1/60th of a second. The numbered blocks show the time periods needed for each frame update. A locked frame-rate waits for the start of each frame-time period to start updating the current frame. If a frame takes longer than the set frame-time, such as Block 1 in the diagram below, then the next frame will be delayed until the start of the next frame-time block.

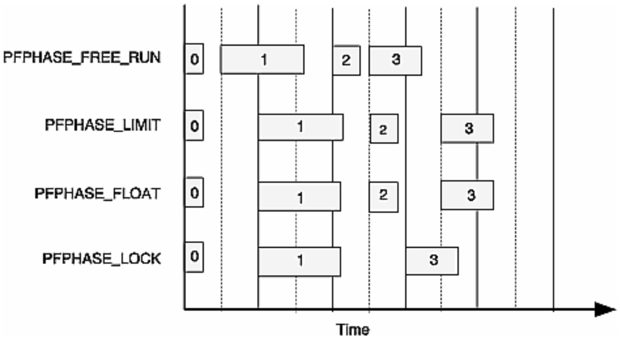

Figure 2 Illustration of frame rate control mechanism in Performer [Eckel 2001].

The implication of this approach is that each frame update is of a variable time length, with the optimal time length being less than a single frame-time period. Decomposing this further, we find that each frame-update is made up of a number of smaller tasks. A single frame update is broken into the following three stages: *object update*, *draw*, and *LOD management*. For this rendering scenario, the objective is not more speed, but an improvement in image quality, based upon non-uniform spatial allocation of triangles. Therefore, we sacrifice some triangle rendering time, to gain visual quality by allocating the majority of triangles to regions that are most visually important.

The following equations relate these stage times to the frame time:

$$T_{frame} = T_{update} + T_{draw} + T_{LOD} \qquad (1)$$

where:

T_{frame} is the total amount of time taken to render an animation frame. This may or may not differ from the frame rate set by the user.
T_{update} is the time taken to update the scene graph with any changes occurring due to user interactions etc.
T_{LOD} is the time taken to calculate LOD values in the stored scene graph.

The object update stage takes care of user input and the positioning of objects in the scene and is dependent upon the number of objects in the scene graph and user interactions. The draw stage encompasses all rendering processes that produce the final image to be drawn. Its processing time is again a function of the number of polygons to be drawn. The LOD management stage calculates the visual importance of visible objects and assigns object levels of detail based upon their visual importance. This stage depends upon the number of objects in the scene and the number of polygons that make up the visible image.

From Equation 1, any increase in the LOD management time will impact upon the overall frame time. The goal of this system is to try to maintain a constant frame time throughout the entire graphical simulation. Therefore, by keeping T_{frame} fixed, any increase in T_{LOD} will result in a decrease in T_{draw}. This implies that there is a sacrifice in the number of polygons rendered from introducing the visual attention based LOD management technique. The aim of this method is to compensate for this loss in polygons by increasing the perceived visual quality of the scene by applying the triangles to where they count. It is believed that the perceived visual quality will be a further improvement upon an unmanaged scene, as the geometric detail will be at the positions regarded by the viewer.

The method for determining the level of detail of each visible object is central to this project, and is dependent upon the calculated visual importance of the object, as shown below. This determines the number of polygons used to represent object *i* based upon the importance of the object and the original number of polygons in the object:

$$P_{LOD}(i) = LOD(I, P(i)) \tag{2}$$

with the constraint that:

$$P_{LOD}(i) \leq P(i) \tag{3}$$

The function *LOD* is evaluated as follows:

$$LOD(I, P(i)) = P(i) + aI + s \tag{4}$$

where:

$P(i)$ is the number of polygons in object *i*;
a is an arbitrary scale value;
I is the object importance $\in$ {-1,1};
s is the stress adjustment.

The stress adjustment value is calculated from the time taken to update the previous frame. It is taken from a value returned by Performer. These equations interact due to the frame rate management enacted by Performer. If the value of T_{frame} gets too large, then the stress adjustment variable *s* is increased to use lower levels of detail, and thus raise the frame rate to that specified by the user. The result of this equation is that the LOD of the object is a single value that lies somewhere between 0 and *#polygons*. Progressive calculations shift the LOD value up and down this range. The value is then scaled to a number that lies in the range [0..#levels]. This value is used by Performer to select the object representation to draw, with a value of 0 being the highest level of detail and *#levels* being the lowest. Level switching occurs at the boundaries of the values, that is, when the integer portion of the value increases or decreases. Due to this fact, additional techniques have been employed to delay level switching, so that an object does not rapidly oscillate between two levels of detail. These techniques are detailed in Section 3.3.

We now describe the visual attention system used to derive the variable *I* in Equation 4.

3.2. The Visual Attention Model

In essence visual attention theory seeks to explain the way humans focus their conscious attention [Treisman 1980]. It is commonly accepted that attention is affected by certain visual attractors as well as predefined tasks [Wolfe 1994]. These concepts are often referred to as *bottom-up* searching and *top-down* searching respectively [Wolfe 1996]. They are grouped under the umbrella of pre-attentive processing , due to the belief that a subconscious process analyses information in the visual field to direct attention to the location of interesting objects [Wolfe 1994]. It is this belief that underpins the new technique, where in simulating this pre-attentive process, it is possible to determine regions of interest in three-dimensional scenes.

Another aspect of visual attention theory is the integration of pre-attentive visual information. It is known that bottom-up and top-down analyses are used to create a ranking of objects in order of their attentional priority or visual importance [Wolfe 1996]. Features are combined with some form of weighting to generate a salience or activation map [Muller 1995]. Attention is thus concentrated at the peaks of the activation or importance map [Wolfe 1994]. This LOD management technique generates such a map and uses it to allocate object levels of detail, with higher levels given to objects with the greatest attentional importance. Figure 3 shows an example of such an importance map, and how it highlights the most visually conspicuous regions of the image.

Figure 3 The image on the left shows a screen capture of an animated sequence. The image on the right shows the corresponding activation map. The brighter areas on the map show regions of higher visual importance [Yee 2001].

The visual attention model must also be able to effectively predict user eye movements, without unreasonably degrading the performance of the real-time graphics engine. Therefore, it has been decided that the following subset of attentive features will be used in the model: *Size*, *Position*, *Motion* and *Luminance*. These features are chosen due to two factors: their ease of acquisition and their major contribution to the conspicuousness of an image region. Of these five features, the first three are absolute for each object and the last two are relative to other objects in the scene.

The second consideration in regards to the visual attention model is how to combine these features to calculate a per-object importance measure. The importance value is a simple yes/no value mapped to {-1, 1}, to fit in Equation 4. Semantically, it says that either yes, the object is visually important, or no, the object is not visually important. It is calculated from the number of importance features as shown below.

$$I = \sigma\left(\sum_{i=1}^{n} f_i w_i\right) \tag{5}$$

where

f_i is the importance of feature *i* $\in$ {0, 1}
w_i is the weight of feature *i*
σ is a transfer function that converts the continuous value returned by the summation to a discrete {-1, 1} value through the use of a step function.

The following describes the features gathered from the scene in order to carry out the above importance calculations:

- Size–the area of the projected screen space of the bounding sphere of the object is compared with the area of the screen itself. If it is greater than some threshold percentage, then the feature is important, otherwise it is not.

- Position–the position of the centre of the projected bounding sphere is found and compared with a rectangular boundary proportional in size to the viewport. If the point lies inside this rectangle then the feature is important, otherwise it is not.

- Rotation Motion–the maximum rotation delta of all axes of the motion of the object is found. If the rotation delta is not above some predetermined threshold, then the feature is not important. Otherwise it is compared with the average rotation of all visible objects. If the rotation delta is greater than a threshold number of standard deviations from the average, then the object is important, otherwise it is not.
- Translative Motion–the velocity vector of the object is projected onto the viewport. If the magnitude is not above some predetermined threshold then the feature is not important. Otherwise, the velocity vector is compared with the average velocity vector of all visible objects. If the velocity vector of the object is greater than a threshold number of standard deviations from the average, then the object is important, otherwise it is not.
- Luminance–the luminance feature is calculated from changes in luminance values within the area of the projected bounding sphere of the object. These luminance values are captured from a feedback buffer projection of low detail geometry [Neider 1993]. If there is a luminance contrast of greater than 1% within this area then the feature is important, otherwise it is not.

This model has been incorporated into the present Performer LOD management scheme in order to produce a testable implementation.

3.3. Incorporation of New Visual Attention Model into Performer

Figure 4 illustrates the basic functionality of the Performer application pipeline for a single pipe system. The pipeline is divided into three primary stages: the *application* stage, the *traversal/cull* stage and the *draw* stage. The application stage is the portion of the program controlled by the user that performs various simulation, input processing and update functions needed by the graphics application. The traversal/cull stage traverses the scene database and performs various tests to determine the visibility of scene nodes. This stage also selects the appropriate node levels of detail to display as well as performing other state management tasks. The final stage issues commands to the geometry pipeline to create the image for display.

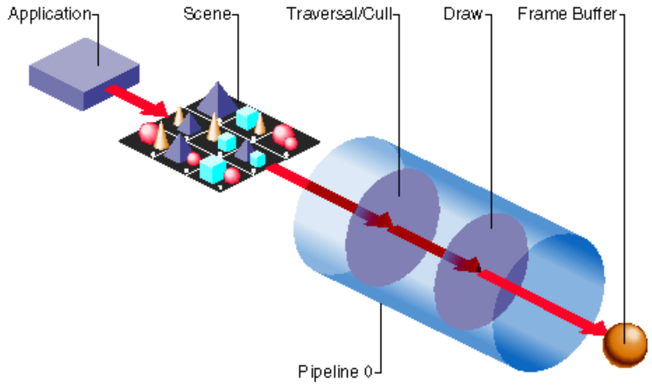

Figure 4 Diagram of major stages of the Performer pipeline [Eckel 2001].

The visual attention based LOD management module is incorporated into this pipeline in the application stage. The Performer documentation dissuades users from including lengthy processing stages in the traversal/cull/draw portion of the pipeline. Therefore, the visual attention LOD management module performs its evaluation of object importance and assigns levels of detail during the application stage. A traversal stage callback function is used to return a single floating point value for each LOD node. This value tells performer which level to draw.

OpenGL Performer LOD Management

OpenGL Performer has sophisticated built in LOD management features [Eckel 2001]. Performer LOD management is closely linked with the frame rate control mechanisms within the system. LOD selection is used to maintain an acceptable, fixed frame-rate as set by the user. The LOD selection is based around the concept of switch ranges. Switch ranges are absolute distances set by a user that specify which object detail level should be rendered. To facilitate this switching mechanism, Performer has a *pfLOD* scene database node that is used to group the object representations. This node contains the switching ranges and other LOD attributes.

Performer processes pfLOD nodes during the database traversal stage of the rendering pipeline. During processing, the appropriate object level is selected based upon its distance from the viewport. Various offsets and scales can be set to manipulate the level selection process. Performer also makes use of blending techniques, in an attempt to mask the popping effect that occurs during level switching.

Figure 5 illustrates the basic LOD management techniques used by Performer. Each car is a different level of detail representation for the single model. The double-ended arrows indicate a switch range for each level of detail. When the car is closer to the eyepoint than the first range, nothing is drawn. When the car is between the first and second ranges, LOD 1 is drawn. When the car is between the second and third ranges, LOD 2 is drawn. When the distance from the eyepoint to the car is within one of the blend zones, the two neighbouring levels are both drawn at complementing levels of transparency.

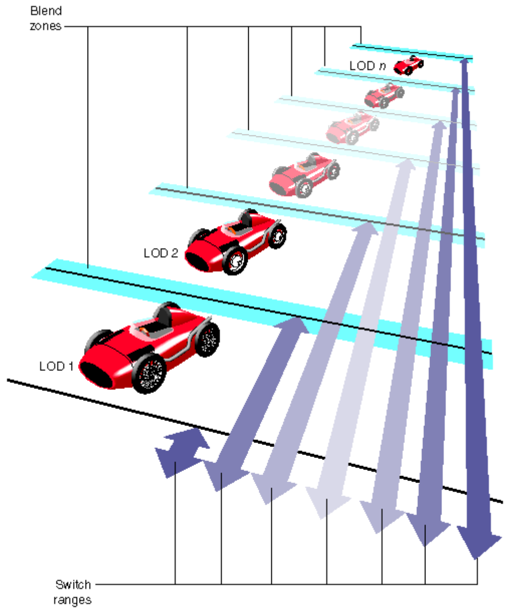

Figure 5 Illustration of LOD management performed nativly within Performer [Eckel 2001].

A final technique used by Performer to facilitate LOD management is a LOD evaluation callback function. This function returns a single floating point value that is used to select the object detail level. This feature is included to provide support for complex LOD evaluations that cannot be implemented using the existing methods. It is this function that implements the previously described visual importance calculations in Equation 5, to assign a LOD value to the object for later rendering. The function gathers the feature values for: Size, Position, Rotational Motion, Translative Motion and Luminance using these values as described to assign either a high or low value of importance to the object. The importance value of the object then influences the final LOD value chosen according to Equation 4.

The above techniques have been implemented within a C++ program using the Performer API as a basis for the system [Eckel 2001] on an SGI 330L Pentium III workstation with an NVidia Quadro2 graphics card, running Linux RedHat7.1. The following images show an excerpt from the test scenes used which illustrates the difference in LOD chosen, biased by the visual importance of the object in question (refer to Figure 6).

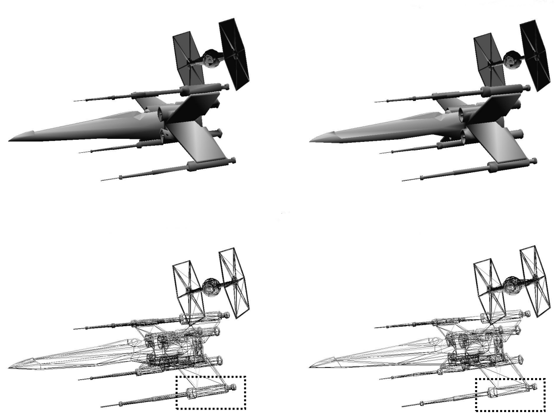

Figure 6 Scenes showing spacecraft from a test animation. The left image is with attention-biased LOD management, and the images on the right are without attention-biased LOD management. The top row is with filled polygons, while the bottom row is in wireframe mode, to show polygonal details.

The X-wing has been selected as a high importance object by the system, and has been given a larger proportion of the geometry within the scene. This is highlighted in the detail views in Figure 7.

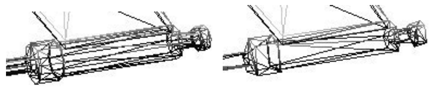

Figure 7 Illustrates detail views from dashed boxes in Figure 6, highlighting the differences in LOD chosen for the objects. The managed object is on the left, while the unmanaged LOD is on the right.

We now describe the assessment of this technique using subjective testing.

4. Subjective Testing

The test for this technique is the subjective impression of the quality of the images generated by the implementation. Due to the lack of an acceptable computational image quality metric, it is often required that image quality be assessed using subjective testing by viewers [CCIR 1994].

There are two main hypotheses we have concerning this visual attention biased approach. First, we wish to test whether the approach is able to improve the appearance of an image using the same number of polygons, by applying the most polygon detail to the locations of visual attention. Related to this hypothesis is the second issue of whether the subjects will notice any differences between the attention-biased and unbiased scenes.

Methodology

The testing technique is using a double stimulus test [CCIR 1994] involving the viewing of a normal LOD image and a visual attention-biased LOD image, in random order. The viewer is asked to assess the quality difference between the images using a continuous scale. The images were viewed under typical office conditions. The 13 subjects were instructed to mark the scale at the point that indicated the difference in visual quality between the two images presented.

Figure 8 Illustration of the three groups of test scenes being under sea with small number of objects (top), jet aircraft with medium number of objects (middle) and spacecraft with many objects (bottom).

Nine test scenes were presented with the number of objects and the amount of motion in the scene varied. The number of objects in the scene was varied over 5, 12 and 20 objects. Within the number of objects, the amount of motion in the scene was varied over low, medium and high amounts. The objects used included, various underwater, jet aircraft and spaceship scenes as illustrated in Figure 8. The two independent variables tested; number of objects and amount of motion, were chosen due to their expected strong influence on the subjective quality of the images. In the low motion, low object number scenes, it was expected that the subjective quality results would differ by a large margin, and so the technique would perform poorly. It was expected that as the number of objects, or the amount of motion increased, the subjective quality results would improve, due to the poorer quality objects not being noticed in the visual noise. Thus, the attention-biased technique was assessed over the expected extremes of its capabilities.

Results

The summarised results are listed in Table 1.

Total	**Mean**	**Std Dev**	**Conf Int**	**T-value**	**Accept H_0**
All	-5	34	20	-0.04	Yes
# Objects	**Mean**	**Std Dev**	**Conf Int**	**T-value**	**Accept H_0**
5	-15	32	19	-0.13	Yes
12	3	36	22	0.02	Yes
20	-3	32	19	-0.03	Yes
Motion	**Mean**	**Std Dev**	**Conf Int**	**T-value**	**Accept H_0**
Slow	-5	34	21	-0.04	Yes
Medium	0	34	21	0.00	Yes
Fast	-9	33	20	-0.08	Yes

Table 1 Summary of results from subjective tests across all viewers.

The results show the mean and standard deviation of the responses of all the subjects. The confidence interval , t-test statistic and acceptance/rejection of the null hypothesis are also listed. The results are grouped into three main sections: overall results, object number results and amount of motion results. The values listed are a millimetre measurement of the distance of the mark made from the centre of the scale used in the questionnaire. Zero thus indicates that the viewers perceived no difference in image quality between the attention-biased and non-biased scenes. A positive value indicates that the viewers considered the attention-biased scene to be better quality, while a negative value indicated that the non-biased image was considered to be better quality.

Discussion

In each case the t-test statistic supported the null hypothesis of no difference between the two images, that is, $\mu_0 = 0$. Furthermore, there were little if no trends shown for the results over the differing experimental conditions. This was contrary to expectations, as we expected the varying numbers of objects to especially influence the values in a strong manner. Therefore, our first hypothesis, indicating an improvement in image quality due to attentional biasing, is not supported by the results. The second hypothesis, that people would not notice an overall difference between the attention-biased and non-biased images, is supported by the results.

However, it should be noted that even though the test statistic was accepted for the number of objects in the scene, in each case, the scene with the lowest number of objects (5) was able to elicit the largest mean value of results (-14.77), even though the test statistic was still well within the acceptance region. This largish mean value was expected, as the lower the number of objects, the more noticeable are any changes in the geometric quality of the objects due to attention biasing.

In addition, the values recorded showed a wide spread, with an overall standard deviation around 33. This indicates that the subjects were unsure of which value to choose, giving support to the notion that the differences are negligible to the viewer.

5. Conclusions and Future Work

In this paper we have presented the conceptual and theoretical basis for an attention-biased LOD management system. We have shown that previous work in image synthesis and image processing has benefited from the application of visual attention principles. We then developed a theoretical basis for the management of LODs within the Performer software system. A visual attention model was then developed which was suitable for use in a real-time graphics rendering system. This approach has then been fully implemented within Performer a simulation system freely available from SGI.

The system was tested with a cohort of viewers. The results showed a lack of improvement in the visual quality of the attention-biased images. However, the attention-biasing mechanism did not produce enough artefacts to lower the quality of the image under this LOD management scheme. This points to potential uses in polygon decimation techniques, whereby polygons may be removed from an arbitrary mesh in visually unattended areas without harming the overall perception of the quality of the scene.

Future work includes the incorporation of this approach into more localised LOD management techniques, where a geometric mesh may have its LOD modified non-uniformly across itself. Application areas for this technique are expected to be found in entertainment, visualisation and simulation systems.

References

BROWN, R., PHAM, B. AND MAEDER, A. 2001. A Fuzzy Logic Model of Visual Importance for Efficient Image Synthesis, *The 10th IEEE International Conference on Fuzzy Systems*, 3, 1400-3.

CCIR 1994. *Methodology for the Subjective Assessment of the Quality of Television Pictures*, Document ITU-R BT.500-6.

CLARK, J. 1976. Hierarchical Geometric Models for Visible Surface Algorithms, *Communications of the ACM*, 19, 10, 547-554.

DE ALMEIDA NEVES, E. M., BORELLI, J. E. AND GONZAGA, A. 2000. Target Search by Bottom-Up and Top-Down Fuzzy Information, *Proceedings of the XIII Brazilian Symposium on Computer Graphics and Image Processing*, Gramado: Brazil, 60-66.

ECKEL, G. AND JONES, K. 2001. *OpenGL Performer™ Programmer's Guide*. CA: SGI Technical Publications.

EL-SANA, J. AND VARSHNEY, A. 1998. Topology Simplification for Polygonal Virtual Environments, *IEEE Transactions on Visualization and Computer Graphics*, 4, 2, 133-143.

FUNKHOUSER, T. A. AND S'EQUIN, C. H. 1993. Adaptive Display Algorithm for Interactive Frame Rates During Visualization of Complex Virtual Environments, *Proceedings of SIGGRAPH '93*, 27, 247-254.

HE, T., HONG, L., VARSHNEY, A. AND WANG, S. W. 1996. Controlled Topology Simplification", *IEEE Transactions on Visualization and Computer Graphics*, 2, 2, 171-183.

HOPPE, H. 1996. Progressive Meshes, *Proceedings of SIGGRAPH '96*, 30, 99-108.

HORVITZ, E. AND LENGYEL, J. 1997. Perception, Attention and Resources: A Decision-Theoretic Approach to Graphics Rendering," *Proceedings of the 13th Conference on Uncertainty in Artificial Intelligence*, San Francisco: CA, 238-249.

ITTI, L., KOCH, C. AND NIEBUR, E. 1998 A Model of Saliency-Based Visual Attention for Rapid Scene Analysis, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20, 11, 1254-1259.

LAMOTHE, A. 1999. *Tricks of the Windows Game Programming Gurus: Fundamentals of 2D and 3D Game Programming*. Indianapolis: Sams.

LUEBKE, D. P. 2001. A Developer's Survey of Polygon Simplification Algorithms, *IEEE Computer Graphics and Applications*, 21, 3, 24-35.

MILANESE, R. 1993. *Detecting Salient Regions in an Image: From Biological Evidence to Computer Implementation*, PhD thesis, Department of Computer Science, University of Geneva.

MÜLLER, H. J., HELLER, D., AND ZIEGLER, J. 1995. Visual Search for Singleton Feature Targets Within and Across Feature Dimensions, *Perception and Psychophysics*, 57, 1, 1-17.

NEIDER, J., DAVIS, T. AND WOO, M. 1993. *OpenGL Programming Guide*. Reading, USA, Addison Wesley.

OHSHIMA, T., YAMAMOTO, H. AND TAMURA, H. 1996. Gaze-directed Adaptive Rendering for Interacting with Virtual Space," *IEEE Virtual Reality Annual International Symposium*, Santa Clara: CA, 103-110.

OSBERGER, W. AND ROHALY, A. 2001. Automatic Detection of Regions of Interest in Complex Video Sequences, *Proceedings of Human Vision and Electronic Imaging VI*, Bellingham: USA, 361-72.

REDDY, M. 1995. A Survey of Level of Detail Support in Current Virtual Reality Solutions. *Virtual Reality: Research, Development and Application*, 1, 2, 95-98.

REDDY, M. 1997. *Perceptually Modulated Level of Detail for Virtual Environments*. PhD thesis, Division of Informatics, University of Edinburgh.

REDDY, M. 1999. Specification and Evaluation of Level of Detail Selection Criteria, *Virtual Reality: Research, Development and Application*, 3, 2, 132-143.

TREISMAN, A. M. AND GELADE, G. 1980. A Feature-Integration Theory of Visual Attention, *Cognitive Psychology*, 12, 97-136.

WATT A., AND WATT M. 1992. *Advanced Animation and Rendering Techniques: Theory and Practice*, Essex: Addison-Wesley.

WERNECKE, J. 1994. *The Inventor Mentor: Programming Object-oriented 3D Graphics with Open Inventor™*, Release 2. Reading: Addison-Wesley.

WOLFE, J. M. 1996. Visual Search: A Review, In Pashler, H. (Ed.), *Attention*, London: University College London Press.

WOLFE, J. M. 1994. Guided Search 2.0: A revised model of visual search, *Psychonomic Bulletin and Review*, 1, 2, 202-238.

YEE, H., PATTANAIK, S. AND GREENBERG, D. P. 2001. Spatiotemporal Sensitivity and Visual Attention for Efficient Rendering of Dynamic Environments, *ACM Transactions on Graphics*, 20, 1, 39-65.