Hierarchical Level of Detail Optimization for Constant Frame Rate Rendering of Radiosity Scenes

S. Nirenstein E. Blake S. Winberg A. Mason

Collaborative Visual Computing Laboratory,
Department of Computer Science,
University of Cape Town.
snirenst@cs.uct.ac.za, edwin@cs.uct.ac.za

Abstract

The predictive hierarchical level of detail optimization algorithm of Mason and Blake is experimentally evaluated in the form of a practical application to hierarchical radiosity. In a novel approach the recursively subdivided patch hierarchy generated by a perceptually refined hierarchical radiosity algorithm is treated as a hierarchical level of detail scene description. In this way we use the Mason-Blake algorithm to successfully maintain constant frame rates during the interactive rendering of the radiosity-generated scene. We establish that the algorithm is capable of maintaining uniform frame rendering times, but that the execution time of the optimization algorithm itself is significant and is strongly dependent on frame-to-frame coherence and the granularity of the level of detail description. To compensate we develop techniques which effectively reduce and limit the algorithm execution time: We restrict the execution times of the algorithm to guard against pathological situations and propose simplification transforms that increase the granularity of the scene description, at minimal cost to visual quality. We demonstrate that using these techniques the algorithm is capable of maintaining interactive frame rates for scenes of arbitrary complexity. Furthermore we provide guidelines for the appropriate use of predictive level of detail optimization algorithms derived from our practical experience.

Keywords: *Rendering, Hierarchical Level of Detail, Hierarchical Radiosity*
Computing Review Categories: *I.3.7*

1 Applying Predictive Level of Detail Algorithms

Level of detail techniques are used to eliminate scene detail selectively and thereby improve frame rates in interactive visualization. Recently *predictive* level of detail algorithms have been proposed, whose aim is not merely to accelerate rendering but rather to fix frame rates to a user defined frequency. These techniques promise to allow strict regulation of frame rates by managing the predicted rendering cost of the scene while optimizing for maximum visual quality. There have been few documented applications of predictive level of detail techniques to practical interactive systems. This paper examines the integration of hierarchical level of detail optimization with perceptually driven hierarchical radiosity scenes.

We present the results of an investigation into the practical feasibility of the Mason-Blake predictive hierarchical level of detail optimization algorithm [8, 9]. Our investigation takes the form of the application of the algorithm to the interactive rendering of hierarchical radiosity scenes. In a novel approach, we treat the recursively subdivided patch hierarchy generated by a perceptual refinement hierarchical radiosity algorithm as a hierarchical level of detail scene description with automatically generated shared object representations. This is the first time, to our knowledge, that hierarchical level of detail optimization has been applied to such scenes in order to provide view-dependent adaptive refinement in real-time.

Traditionally scene descriptions generated by perceptually-driven hierarchical radiosity methods have been rendered at the fixed maximum detail. Subdivision into patches typically results in complex scenes consisting of many times the number of polygons than the original scene. Since this subdivision is closely integrated with the radiosity simulation it occurs as a pre-process to rendering and therefore can make no use of information that is available at render-time about the position and focus of interest of the viewer (ie. it is *view-independent*). By treating the radiosity-generated scene description as a hierarchical level of detail description we allow predictive and dynamic *view-dependent* control over real-time rendering complexity. This allows us to view more complex interactive radiosity scenes at acceptable constant frame rates.

The contributions of this paper are threefold:

- The process of integrating perceptually driven radiosity and hierarchical level of detail control is detailed. This includes a novel benefit heuristic.
- A presentation of the extension techniques which were used to increase the efficiency of the algorithm is given.

- An empirical analysis of how hierarchical level of detail optimization performs in practice.

The remainder of the paper is structured as follows. In Section 2 we discuss previous work, including hierarchical level of detail and hierarchical radiosity. In Section 3 we introduce the application of hierarchical level of detail optimization to radiosity. In Section 4 we describe the experimental system, and in Section 5 we present and discuss the experimental results as well as our extensions to the basic algorithm necessary for its practical application. Finally Section 6 contains some concluding remarks.

2 Background

Predictive level of detail optimization algorithms have been proposed by Funkhouser and Séquin [2], Maciel and Shirley [6] and Mason and Blake [8]. These algorithms are characterized by the fact that they attempt to place an *upper limit* on the rendering complexity of each visualization. By actively regulating the *predicted* rendering complexity of each frame while optimizing for visual quality, they attempt to ensure consistent and reasonable frame rates. The Maciel-Shirley and Mason-Blake algorithms are further distinguished in that they allow the use of hierarchical level of detail descriptions in which shared representations are provided for groups of objects. In addition the Funkhouser-Séquin and Mason-Blake algorithms are *incremental* and so exploit frame-to-frame coherence for efficiency by accepting as input an initial solution derived from the previous frame. As long as the optimal solutions of successive frames are similar, these algorithms are very efficient. Finally, unlike the Funkhouser-Séquin algorithm (counterexample in [7]), the Mason-Blake algorithm provides a solution that is provably at least half as good as the optimal solution (in terms of predicted visual quality) as long as certain criteria are met. The worst-case complexity of the Mason-Blake algorithm for n selected impostors is $O(n \log n)$, but being incremental, its average complexity is dependent on frame-to-frame coherence and is typically much better.

The Mason-Blake algorithm allows the use of hierarchical scene descriptions in which objects are grouped hierarchically into larger group objects which represent the union of their children. A drawable representation of an object is referred to as an *impostor* of that object [6]. The impostors of the children of a node together form a more detailed representation than the impostor(s) of that node. The entire scene consists in its simplest form of a single object, called the *scene object*. Figure 1 shows a simple example.

The Mason-Blake level of detail algorithm is applied once per frame, and its output is a level of detail of the scene object for that frame. Its input is the level of detail selected for the previous frame (if any) and a constant *rendering cost limit* that represents the rendering time available for this frame. The algorithm guarantees that the total predicted rendering cost of the selected level of detail will be lower than the rendering cost limit, while attempting to maximize its perceptual benefit. The algorithm must be supplied with *benefit* and *cost* heuristics that provide reasonable and efficient predictions of the perceptual benefit and rendering cost of potential object representations.

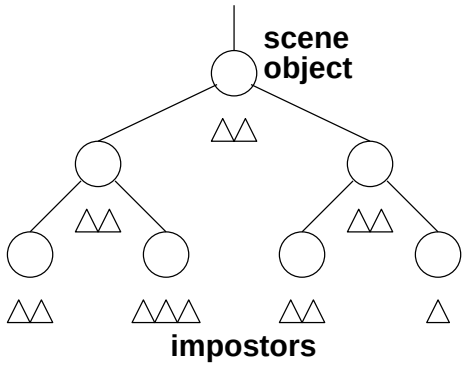

Figure 1: A simple level of detail hierarchy. Objects are represented by circles, and their impostors by triangles. The root represents the lowest level of detail and the leaves the highest detail. For each object, impostors represent various drawable representations in increasing amounts of detail.

3 Level of Detail for Hierarchical Radiosity

Hierarchical radiosity [3, 4, 5] is a physically-based rendering technique in which equations modeling the diffuse transfer of light between surfaces are solved numerically to produce shading intensity values for each surface. As an approximation the scene is modeled by flat polygons, or *patches*, and intensities are calculated for only the vertices of these patches. An initial scene description consisting of a relatively small number of flat *top-level* polygons is adaptively subdivided according to estimates of perceptual importance to produce a final collection of patches that approximate the scene (see Figure 2). The image quality of the resulting visualization therefore depends strongly on the local level of refinement of the patch hierarchy.

For our implementation we use a perceptually-based refinement heuristic as defined by Secchia [10]. This heuristic predicts the visual importance of surfaces according to a model of human visual perception and exploits the exaggerated importance of edges such as shadow boundaries to visual perception. The illuminated patch hierarchies generated using this heuristic are characterized by higher levels of refinement in areas that are, in some sense, perceptually more important. We use the radiosity engine implemented by Secchia to generate input files for our system. Furthermore we make use of the perceptual information inherent in the adaptively refined radiosity hierarchy to exploit visual perception in our benefit heuristic. We use this heuristic to predict the visual importance of potential impostors, taking into account the presence of perceptually important edges as detected by Secchia's refinement heuristic.

Since hierarchical radiosity rendering is performed as

a pre-process to rendering, the adaptive subdivision of the top-level polygons is view-independent. The perceptual refinement heuristic predicts the inherent perceptual importance of patches and can make no assumptions regarding the position or orientation of the viewer. Therefore each part of the scene must be subdivided to the maximum level of detail that might be required in any reasonable viewing situation. Our approach is novel in that instead of simply rendering the entire patch hierarchy at the highest level of refinement reached by the algorithm everywhere in the traditional fashion, we treat the patch hierarchy as a hierarchical level of detail description. The intermediate (non-leaf) patches that were generated and subsequently subdivided serve as low detail impostors for the patches that arose from them. This allows us to choose the level of refinement appropriate for each part of the scene at render time, taking into account the characteristics of the current viewing situation and the rendering time available. Although we refer to the geometric aspects of the hierarchy, the level of details also represent different levels of *illumination* detail.

Our hierarchical level of detail description consists of a hierarchy of nested patch objects. Each patch has a single polygon impostor, and its four children are the patches (if any) into which it was refined. The root object corresponds to the entire scene and has no impostor. Its children are the patches corresponding to the original top-level polygons. The level of detail optimization consists of the selection, for each frame, of a single subtree of the hierarchy rooted at the scene object. The polygon impostors at the leaves of the selected subtree comprise the selected scene representation. By taking advantage of view-dependent information about the position and orientation of the viewer we are able to adaptively and dynamically favour increased patch resolution in areas that are perceptually more important.

Due to the predictive nature of the Mason-Blake algorithm we are able to place firm bounds on the predicted rendering cost of the selected scene representations. The aim is to render, for each frame, the most perceptually effective scene representation that may be rendered in the available rendering time. The point is that reducing rendering complexity in unimportant areas allows us to render more important areas in increased detail. Figure 2 shows example output demonstrating the use of hierarchical level of detail optimization.

It is worth noting that the use of radiosity patches as impostors results in fewer of the "popping" effects that are commonly associated with level of detail rendering, since impostors are always co-planar with the geometry they represent. Some popping effects still occur however. These are the results of sudden changes in detail in areas with a large illumination gradient, such as shadow edges.

4 Experimental System

All tests were conducted on a Silicon Graphics O2[1] workstation. This is a relatively low-end machine by the standards of today, however the usage of such a machine serves as an excellent example of how software techniques may be used extend the utility of such legacy equipment. We expect the algorithm to perform similarly for newer systems, which are similarly matched with respect to the CPU and graphics sub-system. For systems with weaker processing power (relative to the graphics hardware), hierarchical simplification (Section 5.6) may be used to great advantage.

We implemented an experimental system that allows exploration of radiosity scenes both interactively and along pre-defined paths. The system allows the interactive or preconfigured control of various level of detail parameters and views of the scene.

Cost and *benefit* heuristics were provided which predict the rendering cost and perceptual benefit of object impostors. These heuristics were designed to be as simple as possible while still providing acceptable results. The rendering cost of our single 4 sided polygon impostors is measured as a constant 1.2 arbitrary units irrespective of viewing distance and size, i.e. the cost defines a fixed upper limit on the number of polygons selected. We assume that since our polygons are generally relatively small their rasterization cost is relatively small and the rendering cost is therefore dependent mostly on their setup cost. Our results (see Section 5.5) suggest that this is a sufficiently accurate approximation for an O2. For a highly fill-limited graphics sub-system it may be necessary to use alternative heuristic based on solid-angle [1].

The perceptual benefit heuristic was formulated as:

$$\begin{aligned} \text{benefit}(p) &= \text{depthConstant} * \text{depth(p)} \\ &+ \log(\text{sizeConstant} * \frac{\text{area(p)}}{\text{distance(p)}} + 1) \end{aligned}$$

where *area* is the area in object space of the polygon comprising the impostor, *distance* is the distance of the center of the polygon from the viewer, and *depth* is the maximum depth of the full-detail hierarchical level of detail description at and below the node to which the impostor belongs.

We developed this heuristic in an experimental fashion. The *area* and *distance* measures provide an estimate of the projected size of the polygon on the viewport we have found this to be sufficiently accurate. The effect of this is that patches close to the viewer are favoured over those that are further away. The constants determine the relative influence of the terms and can be interactively adjusted in our system. In addition we reduce the benefit of a polygon to zero if it is backfacing or if it is behind the viewer, in order to take advantage of the rendering cost saved by clipping and culling. This experimental heuristic has given us acceptable results, although more accurate heuristics could be developed.

By increasing the depth constant relative to the size constant we distribute the available cost to the patches

[1] 175mhz IP32 MIPS R10000, 128mb RAM

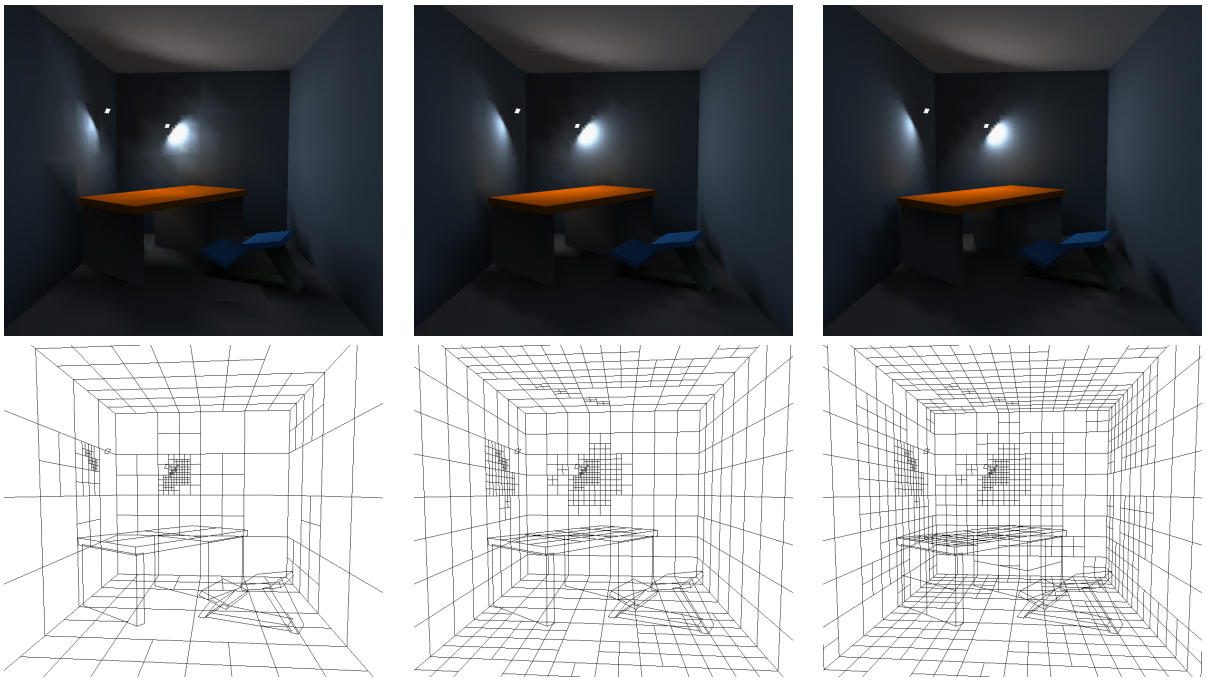

Figure 2: The top three images show the same view of the same scene, with rendering cost limits equal to 500, 1000 and 1500 respectively. At the bottom are wireframe renderings of the same views. Note the adaptive subdivision of patches.

deemed perceptually important by Secchia's heuristic (such as shadow edges). Increasing the size constant relative to the depth constant, shifts cost to those objects nearer the viewer.

5 Results and Discussion

In this section we show how the optimization algorithm depends on frame to frame coherence. We show how this dependence can be reduced and limited in order to achieve constant frame rates. We define *frame rendering time* to be the time taken to render the scene at the selected levels of detail. We further define *frame optimization time* or *algorithm execution time* to be the time taken to execute the level of detail selection algorithm. The *frame generation time* is the time taken to create the frame from start to finish. The frame generation time is simply the sum of the optimization time and the rendering time.

5.1 Dependence of Optimization Times on Changes in Viewing Angle

We begin by investigating the worst case execution times of the optimization algorithm itself. Recall that the optimization algorithm is incremental and exploits frame-to-frame coherence by basing its initial solution on the solution found for the previous frame (Section 2). The success of this approach depends on the degree of coherence between the final solutions of consecutive frames. We therefore measure the change in viewing angle from one frame to the next (for simplicity we disregard the changes in viewing position) along each of several paths through the scene and noted the corresponding optimization times for each frame. We measure these four different *rendering cost limits*. The rendering cost limit dictates how much total detail the algorithm is allowed to select. Figure 3 shows the resulting graphs.

From Figure 3 it is apparent that the algorithm execution time is roughly proportional to the angular change in viewing direction between successive frames. This is to be expected as greater changes in viewing angle result in more objects becoming visible that were previously not visible and vice versa. As objects become newly invisible their allocated rendering cost must be redistributed amongst other objects (some of them newly visible) by means of repeated level of detail incrementations and decrementations.

Although the 8.5 frames per second is not necessarily considered interactive by today's standards, any frame rate may be chosen. A significant increase in the frame rate would lead to a noticible image degredation, however we consider consistency to be more important. A more accurate perceptual metric would provide better image quality with a smaller cost.

Also evident is that the algorithm execution time is roughly proportional to the rendering cost limit. While the aim of the algorithm is to ensure constant *rendering* times irrespective of visible scene complexity, the optimization time (the execution time of the algorithm itself) increases as the amount of detail selected increases. Higher cost limits imply that more selected impostors must be considered for incrementation and decrementation in each iteration of the algorithm.

Since the Mason-Blake algorithm is hierarchical it is able to save optimization time by making use of shared

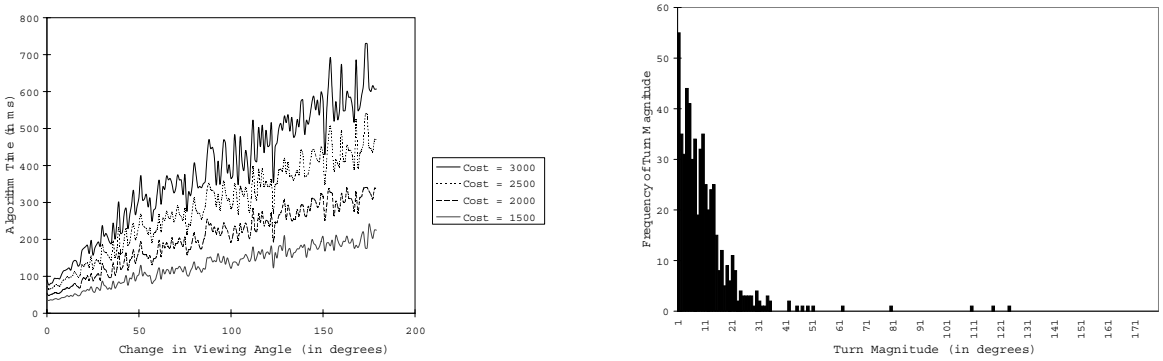

Figure 3: The figure on the left shows that the optimization algorithm execution times for various changes in viewing angle from one frame to the next along a typical path and for various rendering cost limits. The cost of a single impostor is 1.2 units. The "cost" referred to in the diagram is the rendering cost limit. Note that the change in viewing angle is usually less than 30 degrees (at 8.5 frames per second). The figure on the right shows the relative frequency of the changes in viewing angle of a typical walkthrough.

impostor representations that are more efficient to consider than a non-hierarchical collection of impostors providing the same number of levels of detail for each scene object. This saving increases as the rendering cost limit decreases, since lower detail impostors are shared to a greater extent than higher detail ones.

The linear dependence of optimization time on change in viewing angle implies that the algorithm execution times are lower in cases with greater frame-to-frame coherence, as expected.

5.2 Frequency of Turn Magnitudes

Most of the turn magnitudes plotted in Figure 3 represent pathological "non-incremental" cases in which the change in viewing angle is great and there is little coherence from one frame to the next. We therefore measured the relative frequency of turn magnitudes for a typical walkthrough in our system.

Figure 3 shows the resulting graph. It is clear that small changes in viewing angle greatly outnumber large ones, with changes above 30 degrees being extremely rare. We expect this to be the case in any useful interactive visualization system, as very large turn magnitudes are generally distracting to the user and in fact unlikely to occur at all at high frame rates.

5.3 Algorithm Execution Times

The high degree of dependence of the algorithm execution time on frame-to-frame coherence and the relative infrequency of large turn magnitudes (and associated poor frame-to-frame coherence) imply that the average execution time of the algorithm may be somewhat different to the worst case time. Indeed, this is the *raison d'être* of the incremental algorithm: to exploit frame-to-frame coherence and so ensure that average execution times are far better than worst case execution times, at the expense of the efficiency of the worst case. To test this we measured minimum, average and maximum optimization times for a typical path for a range of rendering cost limits.

Figure 4 shows the results. The average optimization time is closer to the minimum time than the maximum, and its behavior is close to linear. We surmise that this is due to the relative infrequency of large turn magnitudes: typically there is significant coherence between successive frames. The minimum algorithm execution time (corresponding to the limit case in which consecutive frames are identical) is essentially constant with respect to the rendering cost limit, but from Figure 4 it appears as if the maximum (approaching the opposite limit in which consecutive frames are completely different) is greater than linear order.

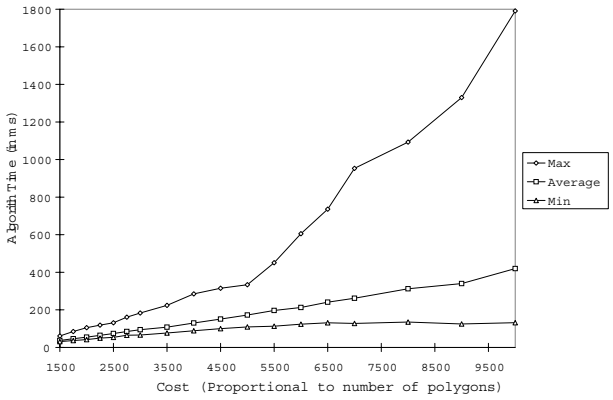

Figure 4: Plot showing how the maximum, minimum and average optimization algorithm execution times (over a typical walkthrough) vary with increasing rendering cost limit. The cost of a single polygon impostor is 1.2 units.

The usefulness of the incremental algorithm hinges on the fact that consecutive frames generally exhibit a high degree of inter-frame coherence, as suggested by the relatively high frequency of small turn magnitudes shown in Figure 3. There are nonetheless cases in which coherence

is limited and optimization times are significantly high. If left unchecked these may lead to excessive inter-frame delays due to the cost of the algorithm itself rather than the actual rendering.

5.4 Constancy of Frame Generation Times

To test the constancy of frame generation times, we measured instantaneous frame rates (defined as the inverse of frame generation time) for each frame of a typical walkthrough, with a rendering cost limit corresponding to 2500 selected impostors (or 2500 rendered polygons). In order to deduce the cause of any irregularities we found, we also measured the frame rendering times and the optimization times for the same walkthrough.

Figure 5 shows the results. It is clear that frame generation times vary dramatically from one frame to the next. We note however that the time taken to render the selected scene representation is relatively constant over all 160 frames, varying between approximately 50 and 80ms. This shows that the algorithm is successful in maintaining relatively constant rendering times. Furthermore, we note that optimization algorithm execution times vary dramatically from one frame to the next, and that there is a marked correlation between the troughs in the graph of frame generation times and the peaks in the graph of optimization times. This suggests that the variation in frame generation time is dependent mainly on variations of the execution time of the optimization algorithm; the algorithm is successful in regulating frame rendering times, but is not guaranteed to take a limited or constant amount of time to do so. The objective thus becomes to place limits on the execution time of the optimization algorithm itself.

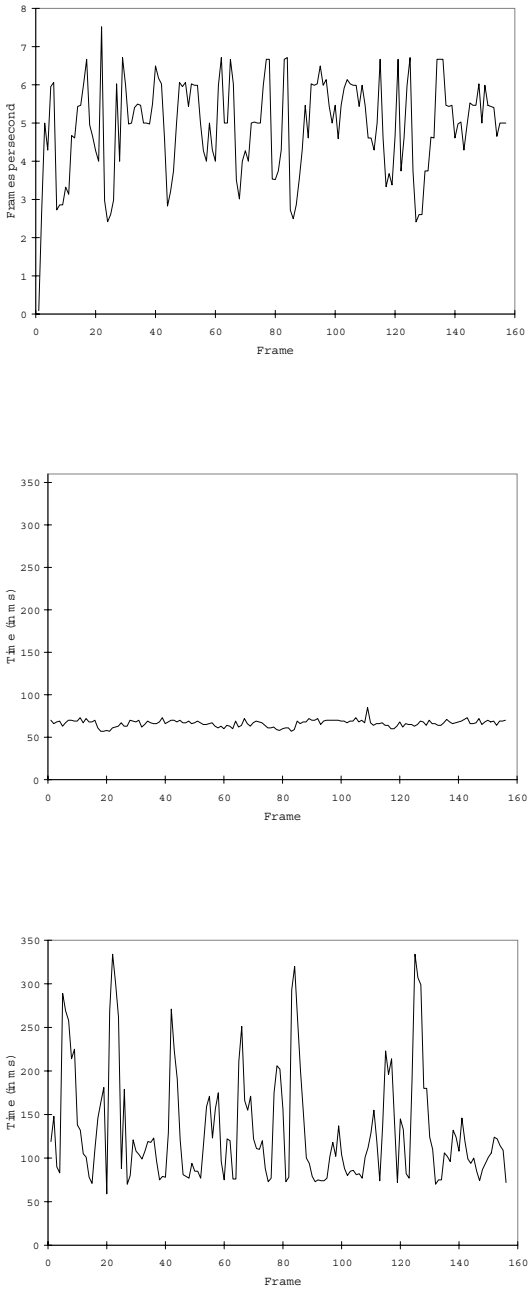

Figure 5: Plots showing (from top to bottom) instantaneous frame rates, frame rendering times (excluding optimization time) and optimization times (excluding rendering time) for each frame over the course of a typical walkthrough. The rendering cost limit is 3000, equating to 2500 single-polygon impostors. Note that the Mason-Blake algorithm is successful in ensuring constant rendering times. However the issue that is discussed in this paper is the non-constant execution of the optimization algorithm itself.

5.5 Truncation of Algorithm Execution

The inconsistency of frame optimization times, if left unchecked, might undermine the ability of the algorithm to regulate frame generation times. We therefore implemented a simple cut-off scheme in which the optimization algorithm's execution is simply halted if its execution time is found to have exceeded some predetermined limit. In the event of the algorithm being halted the solution reached so far is used as the final solution. Due to the iterative refinement strategy employed by the algorithm, its selected solution after any iteration *always represents a feasible and complete* (although not necessarily half-optimal) solution to the hierarchical level of detail optimization problem.

We measured the instantaneous frame rates achieved for a walkthrough with this technique. Figure 6 shows the results. It is clear that time-truncation of the optimization algorithm succeeds in ensuring an essentially constant frame rate, irrespective of visible scene complexity.

The disadvantage of truncating the algorithm execution time is that in the frames where the execution is truncated the algorithm produces a potentially less than half-optimal solution. This may result in occasional drops in

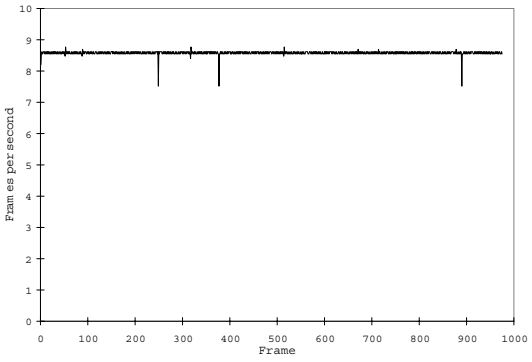

Figure 6: Frame rates of a typical walkthrough (calculated via timing the intervals between frames), with optimization times truncated at 50ms. The cost limit is 1500, corresponding to 1250 single-polygon impostors. The full detail scene representation contains 36879 polygons.

visual quality. The amount of error introduced by truncation is approximately proportional to the amount of time truncated.

Since there is significant coherence not only between successive optimal detail levels but also between the changes in successive optimal detail levels over a series of frames, optimization time skipped on one frame is typically borrowed and then "repaid" in the form of additional computation in the following frames. The error introduced by truncation will always be corrected swiftly as long as excessive execution times are rare. In a typical system the image quality would worsen immediately after a sudden excessive motion by the viewer and then progressively improve (over a few frames) during periods of relatively little incoherent motion.

As we noted with regard to Figures 3 and 4, optimization time is dependent on the rendering cost limit and the degree of coherence between successive frames. Because the average optimization time is closer to the minimum optimization time than the maximum, we can expect the frequency of truncations to be relatively low.

5.6 Hierarchy Simplification

Recall from Figure 4 that the average optimization time was less than 100 ms for rendering cost limits lower than approximately 2500, corresponding to the selection of more than 2000 individual impostors. Because of the nature of our impostors, this corresponds to only around 2000 polygons. This fine granularity of one graphics primitive per impostor represents a worst case for our algorithm, since every single polygon in the scene must be individually considered for selection. In fact, due to the speed of the graphics hardware, the consideration of an impostor for selection may be more expensive than simply rendering it.

To improve this situation we implemented a *hierarchy simplification* strategy in the form of a transformation that reduces the hierarchy by recursively collapsing multiple impostors into single shared representations. After application of this transform, the impostor of each object is the union of the impostors that previously belonged to its children. The leaves of the hierarchy are removed, as their impostors are now replaced by those of their parents. In the instance of our radiosity hierarchy a single application of the transform results in each object (or patch) having a single impostor consisting of four polygons (see Figure 7). A second application results in impostors of sixteen polygons, and so forth. The general effect of the transform is to exponentially increase the granularity of the impostors so that more scene geometry is represented by each impostor. The cost and benefit heuristics must of course be adjusted accordingly.

To test the success of this approach we measured optimization times for a walkthrough of a scene after zero, one and two applications of the hierarchy simplification transform. The results are shown in Figure 7. The rendering cost limit in each case corresponds to a maximum selection of 1666 polygons. The result of applying the simplification transform is to greatly reduce the optimization time required to select the same amount of scene detail. After only one application of the transform the optimization times in Figure 7 are reduced to well below 25 ms for inter-frame turn magnitudes less than 50 degrees and for a selected scene representation consisting of around 416 impostors.

It is important to note that after the application of the transform (and adjustment of the cost heuristic to reflect the fact that impostors are now more expensive to render) *the amount of detail that may be rendered within the available time does not change*. Instead we have traded flexibility of detail selection for speed of optimization, by decreasing the number of possible combinations of impostors from which the algorithm may choose. We have found in practice that a single application of the transform in our case results in an almost imperceptable loss of quality, whereas two or more applications tend to result in visible degradation. Figure 8 compares the visible effects of zero, one and two applications of the transform.

The number of times the transform needs to be applied depends entirely on the system hardware. Once again this is a sacrifice of rendering quality in order to maintain constant, interactive frame rates on low end machines.

5.7 Dependence of Frame Generation Times on Scene Complexity

In order to test the dependence of frame generation times (and therefore frame rates) on the complexity of the full detail scene, we measured non-optimized (full detail) rendering times, optimized rendering times, optimization times and optimized frame generation times for identical walkthroughs of increasingly complex versions of the same scene, with the rendering cost limit held constant throughout.

Figure 9 shows the results. The unoptimized rendering renders the impostors at the leaves of the hierarchical scene description and the unoptimized rendering time in-

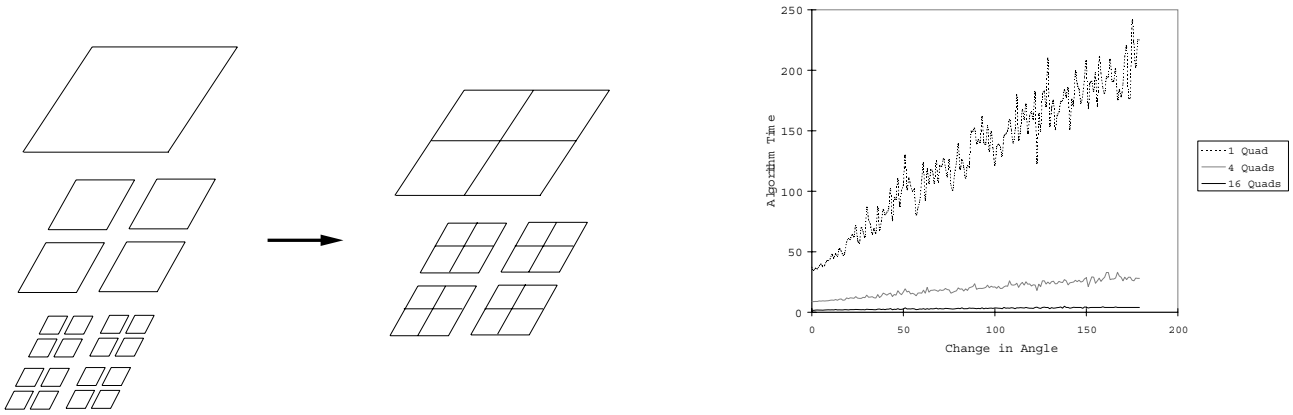

Figure 7: The figure on the left shows the first iteration of hierarchy simplification. Note that an entire level of the hierarchy has been removed. Also note that the visible level of detail has not been adversely affected: The steps between the levels of detail are simply bigger. The figure on the right shows optimization algorithm execution times (averaged over four different walkthroughs of the same scene) for various changes in viewing angle after application of the hierarchy simplification transform zero, one and two times. The rendering cost limit in each case corresponds to 1666 selected polygons.

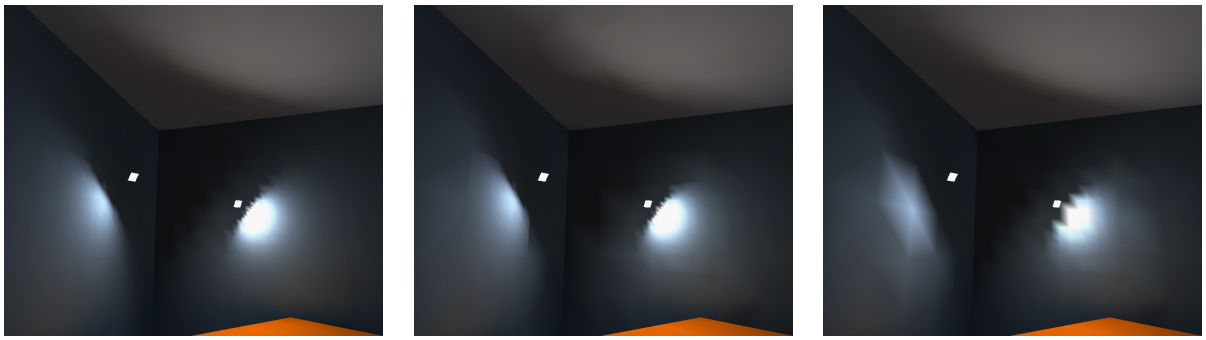

Figure 8: The same view of the same scene after zero, one and two applications of the hierarchy simplification transform.

creases linearly with the complexity of the scene description, as we would expect (since the number of leaf nodes in a regular hierarchy increases linearly with the total number of nodes). The rendering time of the optimized scene is constant irrespective of full detail complexity, as we would also expect since the complexity of the selected scene representation is dependent only on the constant rendering cost limit. The optimization algorithm execution times are constant except for low scene complexities where they increase with increasing scene complexity, probably due to more successful caching of smaller scene descriptions. The frame generation time, being roughly the sum of the optimization time and the rendering time, behaves similarly to the optimization time and becomes constant for increasingly complex scene descriptions.

6 Conclusion

The results presented in this paper attest to the predictive nature of the hierarchical level of detail optimization algorithm, showing that it may successfully be used to ensure fixed frame rates, subject to the accuracy of the cost heuristic used. The algorithm selects a scene representation for every frame that can be rendered in the available time, regardless of the complexity of the full detail scene representation and the complexity of the visible portion of the scene.

The most significant obstacles to the algorithm appear to be the fine granularity (in this case) of the level of detail description and the destabilizing effects of visibility culling on frame-to-frame coherence. Our results show that the Mason-Blake algorithm is capable of maintaining regular frame rendering times, but that the irregular execution times of the algorithm itself threaten to destabilize frame rates unless they are actively controlled. We showed that through the use of a simple cut-off scheme it is possible to ensure that the algorithm execution is not allowed to impair frame rates, with little degradation of image quality.

The irregularity of the algorithm execution times is the result of the algorithm's strong dependence on frame-to-frame coherence. The average performance of the algorithm is far better than the worst-case performance, which arises when frame-to-frame coherence is lacking. The use of frustum culling tends to destabilize the algorithm by creating coherent irregularities in the visibility of groups of

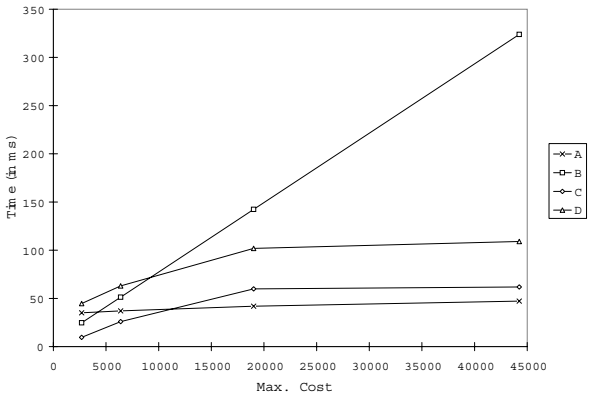

Figure 9: Graphs showing the effects of increasing full detail scene complexity (max cost) on: **(A)** optimized rendering time **(B)** unoptimized (full detail) rendering time **(C)** optimization times **(D)** optimized frame generation times The rendering cost limit is held constant throughout.

objects that cause the algorithm to suddenly redistribute, at some optimization cost, large amounts of rendering resources. The truncation of optimization times serves to resolve this.

The worst-case performance of the algorithm is directly proportional to the sheer number of available impostors, and is therefore strongly dependent on the granularity of the impostor representations. The experiment presented here represents a worst case situation in which every impostor consists of only one polygon. To improve algorithm performance we successfully developed a hierarchical simplification algorithm which increases the granularity of the impostors at small cost to visual quality. As a general rule we suggest that the consideration of an impostor by the level of detail algorithm should always be significantly cheaper than the simple rendering of the impostor. The hierarchical simplification may be used to trade visual quality for efficiency. This is effectively another means to tune the system for low end systems.

The final result of our implementation was a working system in which the Mason-Blake algorithm was used to successfully regulate frame rates while providing acceptable levels of visual quality.

6.1 Future Work

In a level of detail system, it is desirable to have the detail distributed to the most perceptually important parts of the scene. These parts are difficult to quantify, however they are a subset of the parts of the scene which fall into the *view frustum* and are *unoccluded*, namely the *visible* parts of the scene.

A future research direction could be the development of a more perceptually correct benefit heuristic. This would include the visibility factors mentioned above, as well as the direct application of psychophysical perception theory, i.e. the consideration of factors such as color, edges, motion, etc.

References

[1] S. Coorg and S. Teller. Real-time occlusion culling for models with large occluders. *1997 Symposium on Interactive 3D Graphics*, pages 83–90, April 1997. ISBN 0-89791-884-3.

[2] T. A. Funkhouser and C. H. Séquin. Adaptive display algorithm for interactive frame rates during visualization of complex virtual environments. In *Computer Graphics Proceedings Annual Conference Series*, volume 27, pages 247–254. ACM SIGGRAPH, August 1993.

[3] P. Hanrahan, D. Salzman, and L. Aupperle. A rapid hierarchical radiosity algorithm. In *Computer Graphics (ACM SIGGRAPH '91 Proceedings)*, volume 25, pages 197–206, 1991.

[4] P. S. Heckbert. *Simulating Global Illumination Using Adaptive Meshing.* PhD thesis, University of California, Berkeley, 1991.

[5] N. Holzschuch, F. X. Sillion, and G. Dretakkis. An efficient progressive refinement strategy for hierarchical radiosity. *Fifth Eurographics Workshop on Rendering*, pages 343–357, June 1994. Held in Darmstadt, Germany.

[6] P. W. C. Maciel and P. Shirley. Visual navigation of large environments using textured clusters. In *1995 Symposium on Interactive 3D Graphics*, pages 95–102, April 1995.

[7] A. E. W. Mason and E. H. Blake. A predictive incremental hierarchical level of detail optimization algorithm. Technical Report CS99-04-00, University of Cape Town, 1999.

[8] A. E. W. Mason and E. H. Blake. Automatic hierarchical level of detail optimization in computer animation. *Computer Graphics Forum*, 16(3):191–200, August 1997. ISSN 1067-7055.

[9] E. Mason and E. H. Blake. A graphical representation of the state spaces of hierarchical level of detail scene descriptions. *IEEE Transactions on Visualization and Computer Graphics*, 7(1):70–75, 2001.

[10] A. Secchia. Perceptual refinement for hierarchical radiosity. Technical Report CS-00-08-00, University of Cape Town, http://www.cs.uct.ac.za/Research/CVC/techrep.html, 1998.