A Rapid Clustering Algorithm for Efficient Rendering

Gordon Müller, Stephan Schäfer, Dieter Fellner

Institute of Computer Graphics, Technical University of Braunschweig, Braunschweig, Germany

Abstract

Hierarchical radiosity using object clusters greatly improves rendering times and reduces memory consumption of radiosity computations. The key feature of the algorithm is using a hierarchy of object clusters to approximate the energy exchange between surfaces. The cluster hierarchy used for this purpose however, must accurately reflect the actual scene geometry to justify this approach. Bad clusters easily lead to rendering artifacts.
Inspired by the results of our hierarchical bounding volume optimization for ray tracing, we applied the same scheme to a clustering algorithm for hierarchical radiosity. Using an object-oriented framework, the extension of the implementation was straight forward and seems to be promising. Due to the good performance of our hierarchy regarding ray tracing, the same data structure could successfully be used for two things: visibility checks based on ray casting and energy exchange for the radiosity computation.
In this short paper, first results regarding our new clustering scheme will be presented. The properties of the underlying bounding volume optimization give reason for interesting applications enhancing various rendering techniques. Some of these new ideas will be discussed here.

1. Bounding Volume Hierarchies

1.1. Overview

Efficient rendering techniques that handle large scene geometries mostly rely on hierarchical data structures and algorithms for spatial queries at multiple levels of detail. A frequently used technique is the use of hierarchical bounding volumes that separate a scene into sub-graphs. Such hierarchies often describe the distribution of objects in a intuitive way. E.g. architectural scenes are often subdivided into rooms, rooms contain furniture, etc.

Using available hierarchy information obtained from the scene allows for many optimization in the rendering process – some of them will be presented in the following section. One of the still unanswered questions in this area is how to create ’good’ hierarchies that perform well for many rendering algorithms. Often, hierarchical scene descriptions result from the modeling process and are often unpredictable in their effect to a specific rendering method.

Recently, we have presented a bounding volume hierarchy construction algorithm [6] that was primarily developed to be used as a ray acceleration scheme. We have compared this method to many other automatic hierarchy construction schemes and have shown the competitiveness of our algorithm. Interestingly, we have found that the key properties of the generated hierarchies are needed in many different applications. The algorithm

- generates tight bounding volumes
- automatically detects inhomogeneous geometric details and separates distant objects
- is run-time efficient

1.2. How Does It Work ?

Basically, the main idea is to guide the construction process by the use of a cost function to minimize the void area of the bounding boxes enclosing the elementary objects or sub-hierarchies. The scene hierarchy tree is built top-down by recursively subdividing the set of scene objects into two disjoint sub-sets.

Starting from the root node, we sort the objects along all major coordinate axes, where the center of an object’s bounding box serves as sorting key. Based on these sorted lists, we evaluate the potential subdivision positions along each axis for each entry in the respective list by splitting the sorted list of objects into two parts. In contrast to the median cut scheme applied in [3], we don’t have a predefined

Published by the Eurographics Association, ISSN 1017-4656.

subdivision position. Instead, we minimize a cost function describing the approximated costs computing the ray/scene-intersection for a specific subdivision position similar to [2]. By minimizing the cost function over *all* subdivision positions, we obtain an optimal subdivision position which generates two new subdivision entities. The subdivision process terminates for subscenes which contain only a single object.

The hierarchy is efficiently to build (time $O(n \log n)$, assuming random split positions). The key to an efficient implementation is to sort the objects *once* for each coordinate axis in a pre-processing step, since the mutual relative object positions will not change in later phases of the algorithm.

1.3. Example

Figure 1 illustrates the resulting bounding box hierarchy when our method is applied to a large architectural scene containing about 100,000 triangles. The construction time in a single-threaded implementation on a MIPS R10000/250Mhz was only 6 seconds. We are working on an multi-threaded implementation of the algorithm that currently shows linear speed-up up to 4 processors.

Even at a relatively low depth of hierarchy (level 13) the basic arrangements of scene objects become evident and give an impression of the distribution of the objects within the scene. At level 15 individual objects (chairs, table, doors, etc.) emerge, in deeper levels these approximations are further refined. Note that only a small number of boxes compared to the number of scene objects is sufficient to achieve a good approximation of the whole scene.

2. Applications

2.1. Radiosity

Hierarchical radiosity with object clusters [8, 9] was able to reduce the complexity of radiosity calculations from their inherent quadratic nature to a computational complexity of $O(n \log n)$ or even $O(n)$. This speed-up was induced by grouping scene objects into a hierarchy of clusters. To compute the energy exchange inside the hierarchy, a tree traversal is started at the root node with a single link, representing the whole energy exchange. Now, links recursively get subdivided if their accuracy is below a user supplied error threshold. In contrast to this scheme, standard hierarchical radiosity [4] starts with linking all input polygons, thus resulting in quadratic costs during the starting phase.

The critical part of the algorithm remains in building an appropriate hierarchy, typically modeled as hierarchical bounding volumes or k-d trees. Due to [5], hierarchies useful for radiosity clustering should adhere to the following guidelines, otherwise bad image quality (i.e. rendering artifacts) and excessive running time have to be expected:

- fast hierarchy construction time for overall performance
- tight fitting bounding volumes to improve simulation quality
- low branching factor to minimize refinement costs
- support of ray acceleration for fast visibility tests

Comparing the properties of our bounding volume optimization with these suggestions, we found that using the very same hierarchy for ray acceleration and as a hierarchy of object clusters for radiosity would be a promising approach. The integration of both algorithms (hierarchy optimization and radiosity clustering) was done with our object oriented graphics platform MRT [1]. Hierarchy nodes representing scenes and subscenes were treated as clusters. Derived scene nodes, containing regular space subdivisions differed just in their implementation of the ray intersection method from ordinary scene nodes. However, they still contain the complete sub-tree of the bounding volume hierarchy which is needed for accurate energy propagation.

Although further tests still have to be carried out, especially comparisons to other approaches, first results confirm our assumption. The chosen scene structuring scheme is well suited to efficiently simulate the energy exchange in complex environments. We applied our algorithm to an architectural scene of an office building containing differently structured pieces of furniture, textures and light sources. The scene was exported from a modeling and rendering package and was not especially modeled for radiosity purposes. It comprises around 100,000 triangles. A radiosity computation of moderate accuracy finished after 420 sec. The high quality rendering shown in Figure 2 took 47 min on an R10000/250MHz and resulted in about 300,000 triangles.

2.2. Culling

Modern scene graph APIs like SGI Optimizer or Fahrenheit [7] offer support for scene spatialization which is mostly used to create scene hierarchies that apply *culling* techniques on the scene traversal. E.g. if the bounding volume of a subscene lies outside the field of view of an observer, no single object of this subscene has to visualized since the individual objects cannot be visible (*view frustum culling*). Also, if the bounding volume of a subscene is completely occluded by non-transparent objects close to an observer (*occlusion culling*), the whole subscene may be removed from the rendering pipeline. Both technique allow for visualization of huge scenes since the number of visible objects is mostly significantly smaller than the total number of objects.

We think that out construction algorithms generates scene hierarchies that are very suitable for occlusion culling, because separated geometry is detected at a very low level of hierarchy and overlaps are often avoided. Preliminary results show that complex scene can be rendered up to a factor of 3 faster when we use our scheme instead of the spatialization functionality of the SGI Optimizer package.

2.3. Scene Graph Optimization

Another application of our bounding volume hierarchy is the optimization of scene graphs used for approximative rendering. High-end graphics hardware greatly benefits from the use of display lists, where precompiled geometry can be stored. During interactive walk-throughs however, efficient handling of these lists is a non-trivial task. How should the geometry be split into object lists suited for preprocessing ? If the list gets too large, view frustrum culling or even occlusion culling is not useful anymore. If a single part of the large list is visible, the whole list must be displayed, thus wasting rendering time. On the other hand, very short lists easily block the graphics pipeline, due to the overhead involved with the activation of each list.

The hierarchy of tightly fitting bounding volumes could be used, to detect objects of spatial coherence that are generally suitable to be stored in the same list. The probability is very high that several parts of the contained geometry are visible at the same time or, that the complete volume is occluded.

Choosing the appropriate hierarchy depth depends on scene complexity and underlying graphics hardware. This is currently under development in our group and will be implemented into our rendering platform in the near future.

Acknowledgements

We would like to thank 4-e-motions for providing us with the architectural scene. Funding of this project by the German Research Foundation (DFG) under grant Fe 431/1-2 and Fe 431/4-1 is gratefully acknowledged.

References

1. D. W. Fellner. Extensible Image Synthesis, in *Object-Oriented and Mixed Programming Paradigms*, Wisskirchen P., (Ed.), Focus on Computer Graphics. Springer, pp. 7–21, Feb. 1996.
2. J. Goldsmith and J. Salmon. Automatic Creation of Object Hierarchies for Ray Tracing, *IEEE Computer Graphics and Applications*, **7**(5), pp. 14–20, May 1987.
3. A. Gröne. *Entwurf eines objektorientierten Visualisierungssystems auf der Basis von Raytracing*, Dissertation der Fakutät für Informatik der Eberhard-Karls-Universität Tübingen, 1995.
4. Pat Hanrahan, David Salzman, and Larry Aupperle. A Rapid Hierarchical Radiosity Algorithm. In *Computer Graphics (ACM SIGGRAPH '91 Proceedings)*, volume 25, pages 197–206, July 1991.
5. Jean-Marc Hasenfratz, Cyrille Damez, Francois Sillion, and George Drettakis. A Practical Analysis of Clustering Strategies for Hierarchical Radiosity. *to appear in Computer Graphics Forum (Eurographics '99)*, 18(3), September 1999.
6. G. Müller and D. W. Fellner. Hybrid Scene Structuring with Application to Ray Tracing, Proceedings of the International Conference on Visual Computing (ICVC'99), pp. 19–26, Goa, India, February 1999.
7. Silicon Graphics Inc. Optimizer Manual. Technical Report, 1997.
8. Francois Sillion. A Unified Hierarchical Algorithm for Global Illumination with Scattering Volumes and Object Clusters. *IEEE Transactions on Visualization and Computer Graphics*, 1(3), September 1995.
9. Brian Smits, James Arvo, and Donald Greenberg. A Clustering Algorithm for Radiosity in Complex Environments. In *Computer Graphics Proceedings, Annual Conference Series, 1994 (ACM SIGGRAPH '94 Proceedings)*, pages 435–442, 1994.

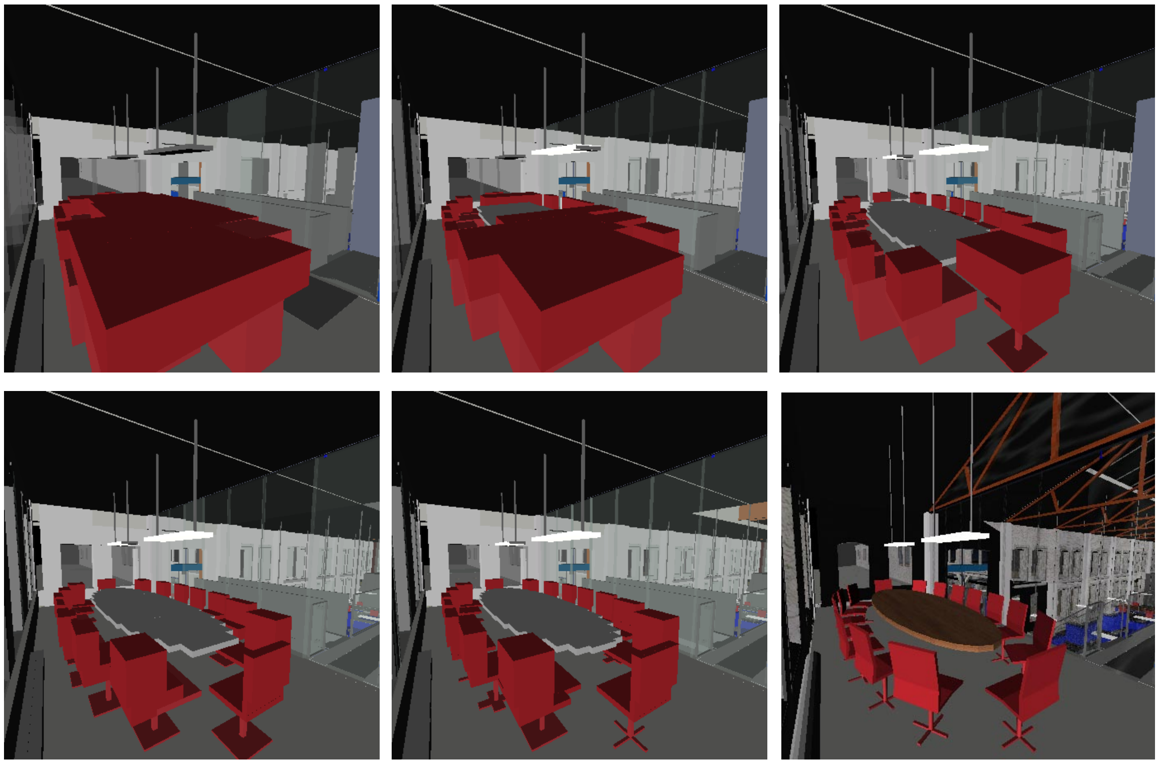

Figure 1: *Visualization of bounding volumes of the architectural scene at tree levels 13, 14, 15, 16, and 20. The image at bottom right shows the Gouraud shaded original scene model.*

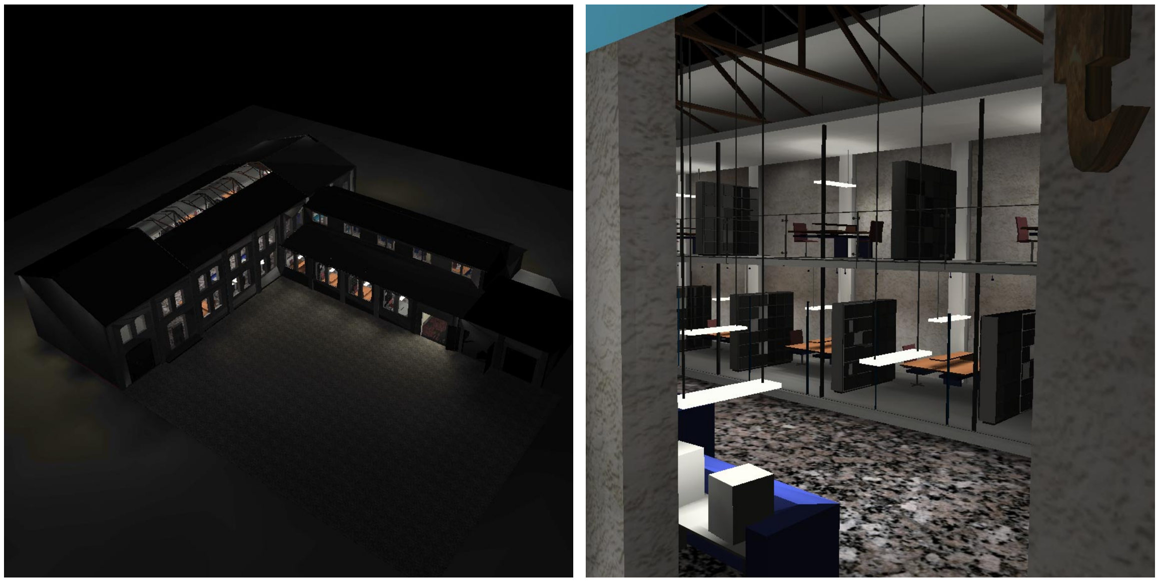

Figure 2: *Radiosity solution.*