A Practical Analysis of Clustering Strategies for Hierarchical Radiosity

Jean-Marc Hasenfratz, Cyrille Damez, François X. Sillion, George Drettakis

EUROGRAPHICS '99 / P. Brunet and R. Scopigno
(Guest Editors)

Volume 18 (*1999*), *Number 3*

A Practical Analysis of Clustering Strategies for Hierarchical Radiosity

Jean-Marc Hasenfratz, Cyrille Damez, François Sillion, George Drettakis

iMAGIS – GRAVIR/IMAG-INRIA, Grenoble, France

Abstract

The calculation of radiant energy balance in complex scenes has been made possible by hierarchical radiosity methods based on clustering *mechanisms. Although clustering offers an elegant theoretical solution by reducing the asymptotic complexity of the algorithm, its practical use raises many difficulties, and may result in image artifacts or unexpected behavior. This paper proposes a detailed analysis of the expectations placed on clustering and compares the relative merits of existing, as well as newly introduced, clustering algorithms. This comparison starts from the precise definition of various clustering strategies based on a taxonomy of data structures and construction algorithms, and proceeds to an experimental study of the clustering behavior for real-world scenes. Interestingly, we observe that for some scenes light is difficult to simulate even with clustering. Our results lead to a series of observations characterizing the adequacy of clustering methods for meeting such diverse goals as progressive solution improvement, efficient ray casting acceleration, and faithful representation of object density for approximate visibility calculations.*

1. Introduction and Motivation

In scenes with great geometric complexity containing hundreds of thousands or even millions of polygons global illumination algorithms require the grouping, or clustering, of the individual primitives. In this way light exchanges can be treated at the level of the clusters and thus the computational complexity of the radiosity solution becomes manageable[14, 12]. Unfortunately, approximations made by hierarchical radiosity algorithms using clustering are very sensitive to the quality of the cluster hierarchy. Due to the complexity of the algorithms and data structures (by definition we are working with very complex models), no experimental analysis of the behavior of clustering algorithms has been undertaken today. Willmott and Heckbert [15] provided an inspiring study for progressive refinement radiosity and hierarchical radiosity without clustering, but evidently this study was restricted in the type of scene considered.

In this practice-and-experience paper, we investigate clustering algorithms for hierarchical radiosity in a practical context. In particular, we have chosen an experimental approach, by comparing the performance of different clustering algorithms. We have concentrated our attention on models provided by real-world applications, in an attempt to uncover problems which real users of radiosity will encounter.

We propose a taxonomy of clustering algorithms, based both on the type of data structure used, and the type of construction algorithm. We then proceed to define requirements for a clustering algorithm. In particular, a clustering algorithm should provide the user with an intuitive time-quality tradeoff: the more time is spent on a solution, the better the quality of the solution. Several other desirable properties are also identified, such as limiting the overlap of clusters, optimizing the "tightness" of the fit of clusters around objects and appropriate size of the clusters with respect to the contained objects.

Once the requirements have been defined, we proceed with a series of experiments run on models used mainly in real-world applications. Various parameters are measured, including the image quality for varying simulation parameters, the cluster construction time, the quality of the hierarchy using different criteria and the speed of ray-tracing for each cluster hierarchy.

The results of these experiments have allowed us to observe a number of interesting properties of clustering, which are discussed in detail. This in-depth study of clustering leads to the understanding that there exists no universal, ideal clustering method, while explaining the relative merits of various approaches.

2. A Taxonomy of Clustering Algorithms

Clustering for hierarchical radiosity was introduced by Smits *et al.* [14] and Sillion [11, 12]. Clustering algorithms can be classified by considering two aspects important to their usage:

1. The choice of data structure used. Broadly speaking, two categories have been presented: regular, typically axis-aligned subdivisions of space and hierarchies of bounding volumes (HBV), which are more closely adapted to the object geometry. Examples of such structures include *k*-d trees and Octrees. Hybrids have also been proposed but have not been used to date in clustering for radiosity.
2. The type of construction algorithm. Again, two basic categories have been used: top-down and bottom-up construction.

2.1. Data Structure Choices

The data structures used for clustering have been mostly inherited from traditional spatial subdivision structures used in graphics.

2.1.1. Octrees and *k*-d Trees

Regular structures such as octrees or *k*-d trees have several advantages:

- They are fast to build (see Section 5.2), and easy to construct since the form of the clusters is (nearly always) predefined.
- They can provide fast ray-tracing since they use traditional ray-traversal mechanisms (see Section 5.2).

Their disadvantages are not specific to clustering for illumination, but due to the rather inflexible nature of their construction:

- Objects which intersect cell boundaries do not have a trivial placement in the tree. As a consequence a heuristic needs to be determined to place the object at an appropriate level.
- The tree can be very deep if the scene contains objects with large differences in scale.
- Since the shape of sub-clusters is prescribed by the subdivision mechanism, many empty clusters can be created. These clusters consume memory and resources since they are considered in all radiosity operations.
- Cluster boundaries do not tightly fit the set of contained objects, resulting in poor-quality estimates of the optical density, for the volumetric estimation of visibility [12].

The first clustering algorithm to use *k*-d trees for illumination was presented by Sillion [11]. This approach uses a traditional, axis-aligned *k*-d tree into which objects are inserted. The strategy chosen for placement of objects intersecting cell boundaries is to put objects at the lowest level entirely containing them. For many models, this can have very negative consequences since a large number of objects can end up at very high levels in the hierarchy, with adverse effects on computation speed. Since the refinement algorithm must, when refining a link to a cluster, create new links for each of its children, a high branching factor may result in long computational times.

2.1.2. Hierarchy of Bounding Volumes

Algorithms based on bounding volumes hierarchies have been used by several researchers [14, 13, 7]. These algorithms use rectangular axis-aligned bounding boxes, and share the following qualities:

Advantages:

- If built correctly, the cluster hierarchy adapts well to the organization of the scene into individual objects (possibly each having its own sub-cluster hierarchy).
- An "intuitive" hierarchy can be produced for a scene with very different object sizes, without creating empty clusters.
- Clusters can be made to tightly fit their contents.

Disadvantages:

- Overlapping clusters are generally unavoidable.
- Bad clusters often result when the scene is considered as a set of individual polygons, without taking advantage of the object structure (*e.g.*, clusters mixing parts of different nearby objects).

Hybrid data structures have also been developed for clustering objects in different domains (notably for ray-tracing acceleration). Cazals *et al.* [2, 3] construct hierarchies of uniform grids, and Klimazewski *et al.* [10] present a similar approach. These methods are however designed to optimize ray-tracing by constructing regular grid structures, and are thus unsuitable "as is" for clustering. Some ideas however, in particular those concerning grouping of objects, by Cazals *et al.* [2, 3], could be applied in part to future clustering algorithms.

2.2. Construction Algorithms

In this section we re-visit the algorithms described above by construction algorithm type. The basic approaches are top-down and bottom-up construction. We will briefly discuss some issues of manual clustering, which is often used in industry and even in research.

2.2.1. Top-down Clustering

Typical top-down construction algorithms include the *k*-d tree (KDT) construction used by Sillion [12]. An initial cell is created, and objects are subsequently added into the cell by appropriately subdividing the cell so that the object "fits" in a sub-cell. As mentioned above, objects crossing cell boundaries are placed high up in the hierarchy.

Christensen *et al.* [4] build a hierarchy of bounding volumes starting with the bounding box of the entire scene. The bounding box is split into eight octants. For each surface contained in this bounding box, if the size of the object is smaller than that of an octant, the object is inserted into the octant containing its centroid. Otherwise, the object is attached as a direct child of the cluster at this level. A new bounding box of each octant is computed, and the algorithm continues recursively in the same manner. In this paper we refer to this algorithm and data structure as TF-OCT ("tight-fitting octree").

2.2.2. Bottom-up Clustering

Bottom-up construction of clusters is inherently more complex, since it requires the examination of the existing objects and their mutual spatial relationships. Since it is in a certain sense an optimization process, the complexities of algorithms suited to such constructions rapidly become quadratic or higher in the number of objects to be processed.

Smits *et al.* [14] mention they use a "modified Goldsmith-Salmon algorithm" without describing the specifics [14]. Sillion and Drettakis employ a hierarchy of regular grids to filter the objects and clusters in order of increasing size, and produce candidate clusters based on spatial proximity [13]. The same algorithm is used by Gibson and Hubbold [7]. In the rest of this paper we refer to this method as PROXI, for "Proximity clustering".

Note that as we choose to group objects based on a minimization function (*e.g.*, minimize the volume of the clusters created with respect to the objects being inserted) [13, 7], an optimal solution requires an exhaustive test of all the combinations of groupings of the objects being considered. Clearly, this expense is extremely costly. A more appropriate grouping approach could be that of Cazals *et al*[3].

2.2.3. Manual Construction

The difficulty of clustering is such that automatic methods are not able to treat all scenes effectively. As a consequence user intervention is inevitable at some stage in the process. Examples of such intervention are the definition of "natural clusters" such as those defined by the group of polygons belonging to a single "object" (a chair for example) or a logical group such as the set of all objects on top of a desk. The special case of touching objects is also important, since it is a way of defining object hierarchies.

Some of these interventions can be handled at input (often the set of objects defining a chair object is defined as a group by the modeling program and then instanced). Such information should be incorporated by the clustering system and used to its advantage when available. Other cases are much harder (*e.g.*, the "objects on the desk" case, or touching objects).

2.3. Improved Approaches

An extended version of KDT, which we call OBT for "Overlapping Binary Tree", was derived as an attempt to merge some of the benefits of Christensen's octree construction and the binary trees obtained with KDT. The algorithm is very similar to the KDT construction, but here, objects crossing the cell boundaries are pushed down in the hierarchy only if the ratio of their size and the considered cell size, is higher than an overlap parameter which can be set by the user (when set to 0, a KDT hierarchy is obtained). The clusters produced by this algorithm can be larger than those of KDT, and therefore may overlap, but the parameter provided gives us control on the maximum potential overlap between adjacent clusters. Specifically, for a given value of the overlap parameter λ, the following relations are true:

- maximal size of the KDT cell to which an object of size S is assigned:

$$x = \frac{2S}{\lambda}$$

- maximal overlap between two adjacent OBT clusters of original size d:

$$o = \frac{8}{3}\lambda d$$

These relations ensure that large clusters will not be overwhelmed by large (and spatially sparse) collections of small objects.

As a consequence, the OBT algorithm produces a deeper hierarchy than KDT, and thus even more empty clusters. In order to get rid of empty clusters, we developed another algorithm that uses an auxiliary OBT hierarchy to build HBV clusters in a second pass. In this second step, we keep only the bounding box of the contents (objects and child clusters) of each non-empty OBT cluster. Therefore, each cluster in the obtained HBV hierarchy will have at most two child clusters. We call the resulting new structure and algorithm OKDT (Overlapping KDT).

Unfortunately, OKDT proved to be unable to separate objects formed of thin, long polygons, such as cylinders, because they were unlikely to be inserted deep in the hierarchy. As a consequence, several clusters contain too many surfaces. This had adverse consequences on ray-casting and refinement time (average branching factor being central for hierarchical algorithms). Therefore, in order to improve our algorithm we also tried applying a PROXI clusterisation process on each cluster containing more than a dozen polygons. We call this algorithm OKDT-P (Overlapping KDT with Proximity second pass).

3. Requirements for a Clustering Algorithm

The most important goal for a good clustering algorithm is to group the objects in a way which represents light transfer to or from the group in the most faithful manner. At the heart

of the hierarchical radiosity algorithm is the assumption that it is possible to replace a “complete” calculation by a simpler one, performed using simplified representations such as clusters. This idea can only be exploited if the resulting simplification only introduces modest perturbations in the calculation.

Unfortunately, a precise definition, or even a set of quantitative yardsticks allowing us to evaluate the quality of such a representation do not currently exist. Instead, a set of heuristics have been developed by various researchers in this domain (*e.g.* [4, 13, 7]). Considering the existing body of work, we can identify two sets of desirable properties for a clustering technique: those affecting the quality of the simulation, and those affecting the overall efficiency of the applications.

3.1. Requirements Regarding the Quality of the Results

We outline below some of the required properties in order to arrive at a satisfactory simulation of light transfer:

1. Monotonicity with respect to light transfer precision. If a light transfer previously represented at a certain level becomes represented at a finer level of the cluster hierarchy, the precision of the light transfer should be increased. If we consider the Time-error graph obtained by plotting the computation time as a function of the solution error, for different tolerance thresholds, we would like to obtain a *smooth and monotonic* function.
2. Overlapping clusters should be avoided as much as possible. Overlapping is problematic mainly because it implies the treatment of the transfer of light of a volume to itself, which is difficult to represent and to express in terms of the hierarchical radiosity formalism. This is especially true in the case where error bounds are estimated to drive the hierarchical refinement.
3. The nature of object group shapes should be preserved as much as possible. Clusters which contain large regions of empty space and scattered small objects should be avoided. Although this may seem evident, many automatic clustering algorithms have trouble respecting this requirement.
4. Objects should always belong to a cluster of “appropriate" size, with respect to their own dimensions. This requirement is difficult to quantify, but is especially important in the context of approximate visibility calculation, where the attenuation of light passing through clusters is estimated based on a volumetric analogy. This analogy relies on the calculation of an optical density for each cluster, which is only meaningful for clusters with well-distributed (i.e. "random") collections of similarly-sized objects [11].

3.2. Efficiency Considerations

Considering the major impact of cost considerations on the usability of clustering radiosity systems, particular attention must be paid to the two following aspects:

3.2.1. Building the Hierarchy

First, we are of course concerned about the efficiency of hierarchy construction. Even though the actual clustering phase is generally a preprocess, and can sometimes be stored with the model, it is still preferable to have efficient construction algorithms:

- Fast cluster construction is important in the modelling/design stage where the model changes significantly and thus long clustering times hinder lighting experimentation.
- For extremely large models it may be impractical to store an additional high-overhead data structure describing the cluster hierarchy.
- Finally, the application of hierarchical radiosity with clustering to dynamically changing environments [5] may require at least a partial rebuild of the cluster hierarchy at interactive rates.

3.2.2. Acceleration of Ray Casting

Another important problem is the potential use of the cluster hierarchy as a supporting data structure to accelerate ray casting queries. Such queries are used for image generation or more often, in the case of radiosity, for visibility calculations when computing form-factors. Indeed, ray casting is the method of choice for visibility estimation in hierarchical radiosity, because the hierarchical algorithm fragments the calculation into a large number of individual queries, each relative to a different emitter/receiver pair [9]. Global approaches such as hemi-cube calculations are therefore inappropriate for hierarchical radiosity.

The requirements of ray-tracing acceleration are often contradictory with those of clustering for illumination. For example, structures which minimise the number of intersections on average in a statistical sense, such as the algorithm of Goldsmith and Salmon [8], result in clusters which contain elongated bounding boxes containing large empty spaces. These clusters are inappropriate for light transfer estimation or optical density estimation, as outlined above.

A possible solution is the creation of two separate structures, one for clustering and one for ray-tracing acceleration. This however would be undoubtedly far too expensive in memory for very large models. In practice, we have observed that the performance cost implied by the use of the clustering structure for ray acceleration is most often acceptable. Some comparative results concerning the performance of various cluster hierarchies as ray tracing accelerators are presented in Section 5.

4. Experimental methodology for evaluation clustering algorithms

4.1. Methodology

As stated earlier, clustering is necessary to make the illumination computation of industrial scenes tractable. Its most

important drawback is that it makes error control very difficult, where it would be necessary to allow fast solutions to be calculated with an acceptable precision. Given the previously described clustering algorithms, we need to determine the critical parameters for their behavior.

Until now, most of the scene models used in the literature were designed by researchers for research purpose. We feel that performing our study on such scenes would have in some way "hidden" most of the problems involved by clustering. In raw “industrial” scenes, poor quality initial meshing, dense distribution of objects or randomly ordered polygons can make clustering algorithms barely usable.

Thus, our approach has been to perform a sufficient number of experiments on a set of complex, "real life" scenes in order to understand and compare each algorithms behavior in terms of quality-versus-time tuning. We limited the range of our experiments to fairly coarse and fast calculations where the impact of clustering is significant.

4.2. Clustering Strategies Studied

The following table summarizes the set of clustering strategies considered in this paper.

PROXI: Proximity cluster. Bottom-up construction after size filtering.

OKDT: Overlapping *k*-d tree. Top-down allowing partial overlap.

OKDT-P: Overlapping *k*-d tree with limited branching. OKDT modified to re-cluster cells with many children, using PROXI locally.

TF-OCT: Tight-fitting octree. Top-down, layer-by-layer octree construction, with re-fitting of octree cell before subdivision.

We did not submit the KDT algorithm to our tests because our experience with it proved that its average branching factor was far too high for it to be usable with scenes as large as the one we used.

4.3. Test Scenes Chosen

We performed our tests on four different scenes, shown in Figure 1. Three of them are rather large industrial-type models while the fourth one is provided as a comparison to show that clustering usually behaves very well when applied to small scenes designed for research purposes. We have taken these scenes as representative scenes for the clustering problem, allowing us to identify the different problems of the algorithms which we will test.

AIRCRAFT (184,456 polygons)
Model of an aircraft cabin (courtesy of LightWork Design Ltd). All objects have been tessellated into (rather small) triangles to account for the rounded shapes.

VRLAB (30,449 polygons)
A virtual reality lab with two floors and mostly overhead lighting (courtesy of Fraunhofer Institut für Graphische Datenverarbeitung). This scene has a mixture of large polygons, likely to be subdivided, and very small patches (on chairs and desktop computers).

CAR (216,157 polygons)
A model of a car interior with very small details, lit by a single overhead console fixture (courtesy of BMW).

OFFICE (5,260 polygons)
A model of a simple office scene. This model is much smaller than the other three and is provided to show that clustering performs well on this kind of small scene usually found in the literature.

4.4. Tests Performed

For each of these scenes, and for each clusterizer we decided to run the following experiments, designed to measure key aspects of clustering:

- **Data structure quality**: We listed the number of clusters in the hierarchy, as well as the average number of surfaces and clusters in each cluster node. These figures are needed to estimate both the memory cost of the hierarchy (compared to the cost of the actual geometry of the scene) and its efficiency for hierarchical radiosity. Recall that the computation speed of cluster based hierarchical radiosity depends on the average branching factor because the refinement algorithm must, when refining a link to a cluster, create new links for each child of this cluster.
- **Measure of time needed to build the cluster hierarchy.**
- **Solution quality**: To evaluate image quality we chose to use a "visual quality" error evaluation on images obtained for a given set of viewpoints rather than a view-independant error metric since we want to study the visual quality rather than the accuracy of the energy transfer quantities. Images were compared to reference images (resulting from a maximum precision calculation) using the following metric: we transform all pixels from RGB space into chromaticity space XYZ. The global image error is then the L_2 norm of the pixel-by-pixel difference image between the current and the reference image, evaluated in CIELUV space [6].
- **Algorithm usability**: To study the "quality versus time" behavior of each algorithm, we rendered each scene using each clustering algorithm, changing only the parameters controlling our refinement process (we used a BF-like refinement algorithm [9], and an error-bound based refinement, both of them showed similar behavior on clusters), measuring the time needed and evaluating the resulting image quality using the method previously described. Since we are interested in the effects of the cluster hierarchy, we do not perform tests on very precise solutions, which involve only surface-to-surface energy exchanges. Instead, we chose our refinement parameters in such a way that the proportion of energy gathered through links

AIRCRAFT

VRLAB

CAR *(courtesy of BMW)*

OFFICE

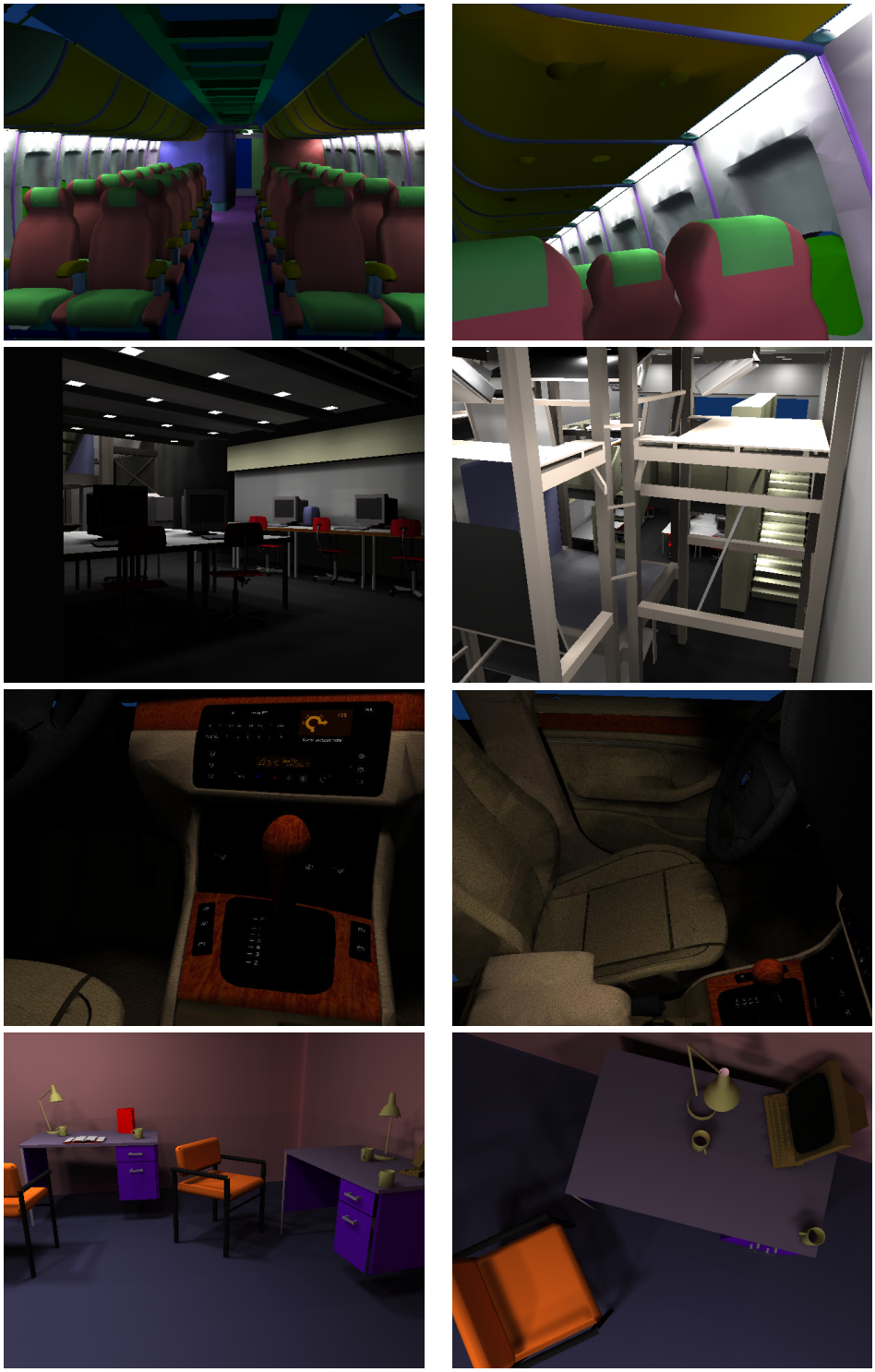


Figure 1: *The four test scenes.*

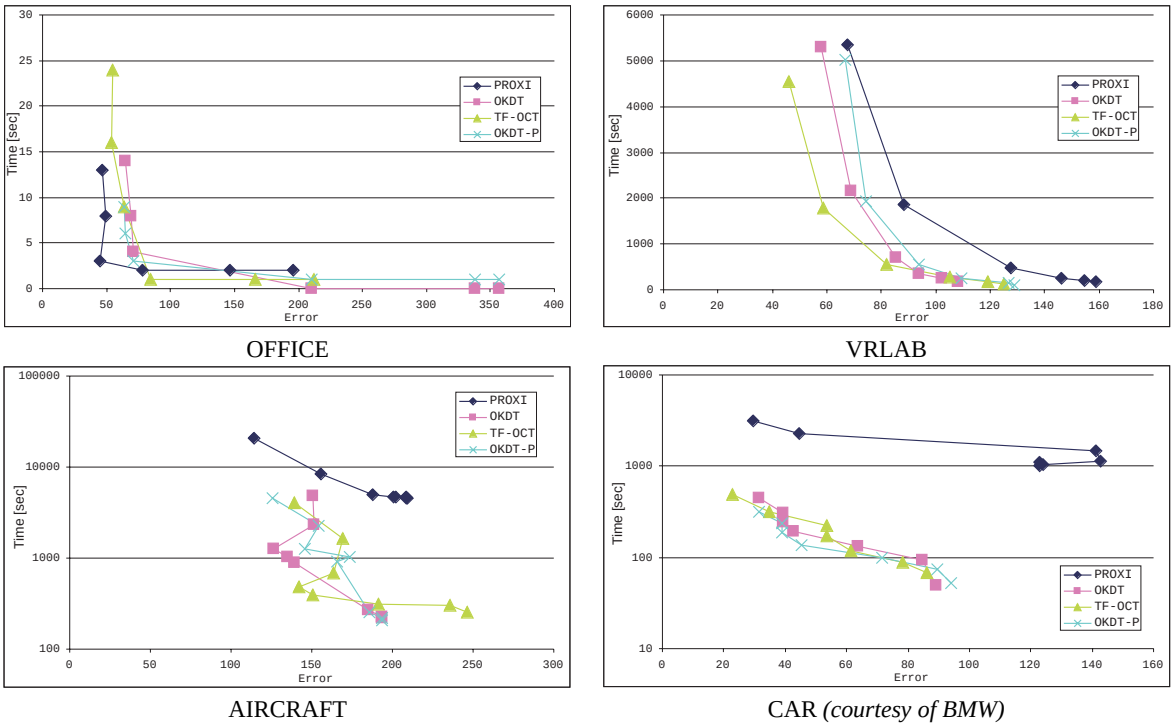


Figure 2: *Time-error curves*

involving clusters is significant (most of the time varying between 30% and 100%).

- **Ray casting acceleration**: As we said in Section 3.2.2, we have seen that it is desirable to use the cluster hierarchy as an acceleration structure to answer visibility requests (see Section 3.2). We decided to test the efficiency of each structure for ray-casting. Therefore, we chose 100,000 random pairs of surfaces in each scene, and measured the total time needed to search for a possible occlusion between each pair.

We also ran a separate set of experiments to evaluate the efficiency of the cluster hierarchy when using approximated volumetric visibility [12]. This alternative to classical exact visibility computation accelerate rendering times when precise shadow area determination is not needed. Instead of the previously defined scenes, we used a model of several trees (Figure 5) representative of some applications where an average representation of light transfers may be sufficient. It is also a good test case for the volumetric visibility algorithm: the set of leaves being a good approximation of a turbid media. We calculated the images using each clustering algorithm and then compared the difference with a reference image obtained using exact surface visibility.

5. Results

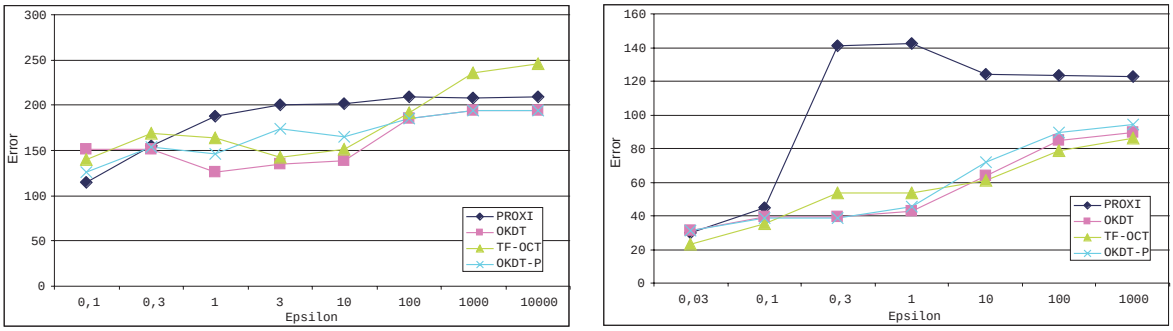
In this section, we present a number of observations drawn from the tests explained previously. We begin with an analysis of the quality aspects of the simulations, looking at the evolution of our error measure with the user-defined error tolerance. We then consider in more detail the capacity of different clustering techniques to assist visibility calculations, with a particular emphasis on computational efficiency.

We ran all algorithms on a Silicon Graphics computer (MIPS R10000 at 250 MHZ) with 4GB of memory.

5.1. Evolution of Solution Error

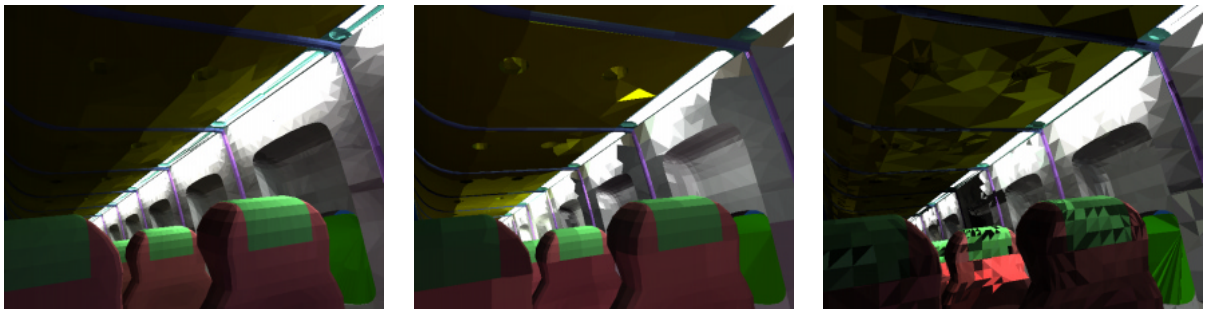
Recall from Section 3 that a major demand on the clustering mechanism (together with the chosen refinement strategy) is that the evolution of computation time and solution quality should be regular and monotonic as the user changes the error tolerance (Figure 3). Ideally, this would result in a very regular and monotonic time/error curve. Such curves are presented in Figure 2 for the four test scenes.

Looking at the four Time-error curves, we see two very different types of behaviors: for the OFFICE and VRLAB scenes, all curves are regular and fairly monotonic, whereas for AIRCRAFT and CAR the variation of error is more erratic. Indeed, looking at a plot of error as a function of the



AIRCRAFT Error CAR Error

Figure 3: *Error curves as function of user-supplied tolerance*



Reference TF-OCT PROXI

Figure 4: *Example solutions exhibiting different (visual) forms of error (see also color plate).*

user-supplied error tolerance, we find that the error in the solution does not always decrease as we reduce the tolerance. Because of limited space, we present only the error plots for AIRCRAFT and CAR (Figure 3); the corresponding curves for OFFICE and VRLAB are monotonically decreasing.

Why can the error increase when we decrease our tolerance? this unfriendly behavior occurs when links refined as a result of the error tolerance change produce a less accurate representation of radiosity exchanges. This is largely a question of refinement criteria, but is also influenced by the clustering strategy, as well as the distribution of objects in the scene. We observe that our scenes can be classified into two types: AIRCRAFT and CAR consist of many small polygons, because of a previous tessellation of the objects. VRLAB and OFFICE, on the other hand, contain objects of varying size, from large walls to small furniture components. Based on our experience and the results of the above experiments, we observe that clustering algorithms have more difficulty with the first type of scene (“polygon soup”). This is especially true of AIRCRAFT because 3D space is very densely populated, resulting in many interactions between clusters which are not separated by a significant distance. These interactions also present a particular challenge to the refinement criterion.

As an illustration of the typical errors created in an approximate solution for such scenes, consider the images in Figure 4. The two approximate solutions have a similar error under our measure, yet they appear quite different visually. The solution using TF-OCT clusters exhibits marked radiosity variations along axis-aligned boundaries, corresponding to the octree cells; on the other hand, the solution using PROXI shows a high variance of radiosity and a speckle pattern, due to the fact that nearby small objects can belong to many different and overlapping clusters, with markedly different radiosities.

5.2. Performance and Visibility Calculation with Clusters

We now consider the performance behavior of the clustering strategies in terms of construction time and as auxiliary structures for visibility calculations.

Construction Time

As explained in Section 2.2, bottom-up construction is a very expensive process since it amounts to an optimization procedure.

Observed computation times for the construction of the

cluster hierarchies support this prediction: the PROXI clustering strategy construction is always much slower than OKDT and TF-OCT, which operate top-down on simple recursive subdivision schemes. These two techniques always take less than 1% of the PROXI time. Interestingly, OKDT-P takes between 20% and 80% of the PROXI time, depending on the distribution of objects in the scene.

Approximate Visibility Calculations

The acceleration of visibility calculations using *equivalent extinction properties* of the clusters has been proposed by Sillion[12]. In this approach, the transmittance factor between two points in the scene is evaluated by considering the entire segment between the points, and its intersections with all clusters, then combining the corresponding attenuation values, in an analogy with partially absorbing volumes. This method, also adopted by Christensen *et al.* [4], is often faster than true ray casting using the surfaces, because of the smaller number of clusters and the ease of computation of ray-cluster intersections.

We computed approximate visibility using clusters in a scene dominated by direct lighting (from the sun), as shown in Figure 5. In this case, the shadow pattern on the floor is essentially an "X-ray image" of the cluster hierarchy, which greatly helps in the comprehension of the cluster distribution.

We observe a clear hierarchy in terms of shadow quality, in the order PROXI (best, notice high quality of trunk shadows), OKDT-P, OKDT, TF-OCT (poorest). Please see images in color section. This is consistent with the intuitive notion that PROXI starts from the objects and build clusters bottom-up, thereby building clusters that are very tight around the objects.

OKDT (and TF-OCT even more so) exhibits some incorrect shadows of large, blocky clusters, due to the constraints in the spatial subdivision. In this respect, OKDT-P effectively improves on OKDT, with a better fit around the objects and more precise shadows.

Ray Casting Acceleration

Interestingly, the computation times shown in Figure 5 increase with the quality of the shadows. This is consistent with the general observation that hierarchical structures with lower branching factors have more hierarchical levels and perform better for ray tracing acceleration.

This reasoning is supported by the analysis of cluster statistics on our test scenes. Figure 6 shows the variation of the following three quantities with the clustering technique, for each test scene:

- total number of clusters
- average number of child elements per cluster
- performance of ray tracing acceleration. This is measured by shooting a large number (100,000) of random rays through the scene and computing ray-surface intersections.

We first observe an obvious inverse correlation between the total number of clusters and the average number of children. In addition, TF-OCT has the largest branching factor for the cluster hierarchy because of its octal subdivision scheme. OKDT also has a fairly high number of children on average, because its construction mechanism offers no way to control this branching factor. Conversely, PROXI has a built-in mechanism limiting the number of children of any given cluster (this operates by grouping objects into overlapping sub-clusters). Therefore it exhibits the lowest branching factor. OKDT-P is intermediate, as expected, because by construction it avoids clusters with many children, handing them to the PROXI clusterizer. Still it avoids the overall large number of clusters of PROXI. A consistent best performer in terms of acceleration is therefore OKDT-P.

Finally, we note the conflicting nature of the two desires for (a) efficient ray tracing acceleration and (b) suitability for radiosity calculations (compare Figure 6 and Figure 2).

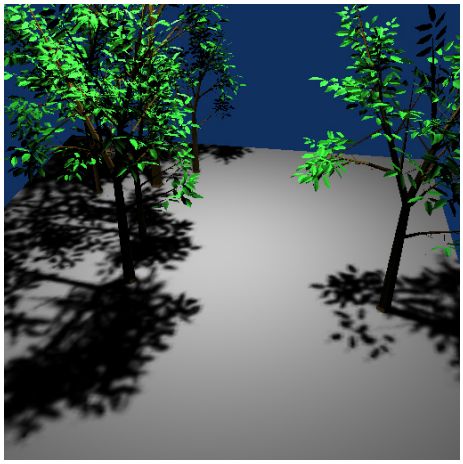
6. Conclusions and Future Work

We have presented an experimental analysis of clustering algorithms for hierarchical radiosity. A taxonomy of clustering algorithms was proposed, followed by a set of requirements for a good clustering algorithm. Guided by these requirements, we developed an experimental methodology based on an image-space quality measure. Extensive tests were run on scenes for the most part developed in real-world application contexts.

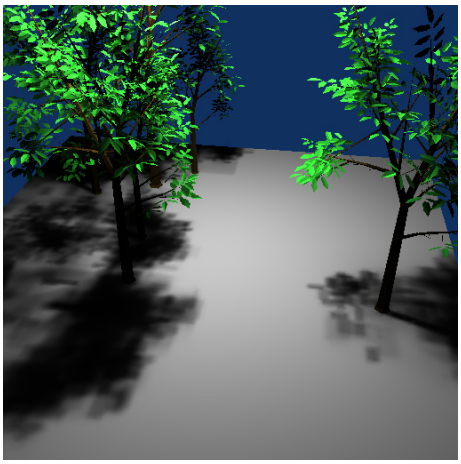
Drawing concrete conclusions from experimental tests such as those performed here is always a delicate task. Nonetheless, there are certain elements which we believe are clear enough to be singled out:

Clustering works well in many cases: in particular, for scenes containing objects of different sizes and a sufficient number of large initial surfaces (walls, floors etc.), all the clustering algorithms tested appear to perform well. The Time-error graphs are smooth and monotonic for these cases, presenting the user with an intuitive time-quality tradeoff.

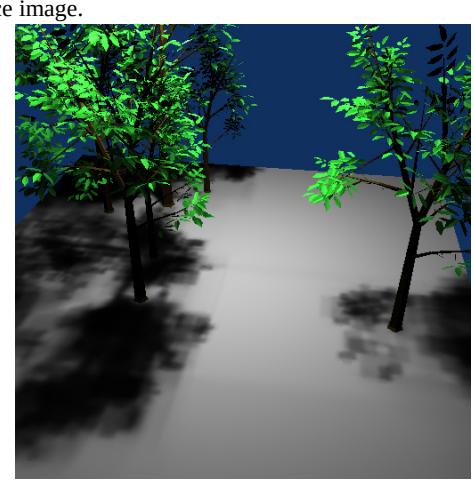
For scenes containing many small objects ("polygon soup"), existing clustering algorithms are less well-behaved. In particular, more time spent computing a solution does not always result in higher quality (see Section 5.1). This is even more troublesome since the scenes in question are typical of industrial "real-world" models, which are often the result of a fine tessellation of some unspecified and unrecoverable modeling format. It is clear that a new approach is required to treat such models, in order to build a hierarchy that follows the definition of objects. Reconstruction of individual



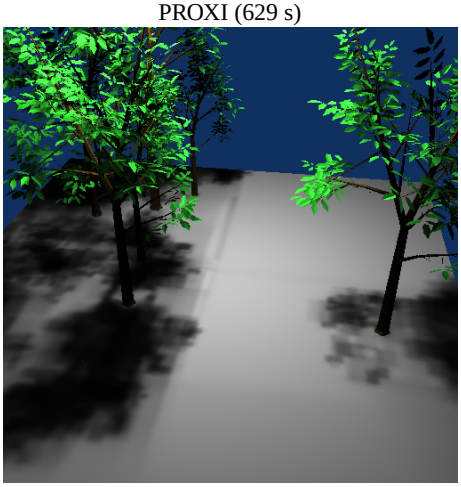
Reference image.



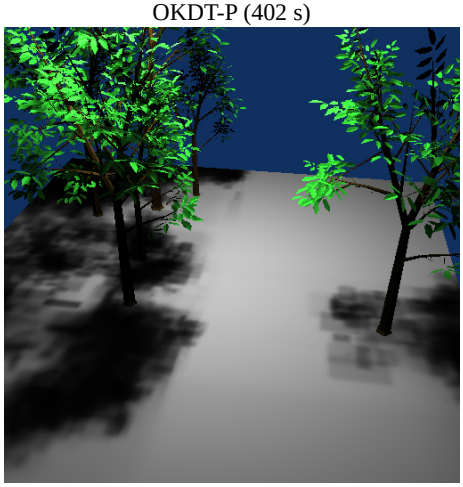
PROXI (629 s)



OKDT-P (402 s)

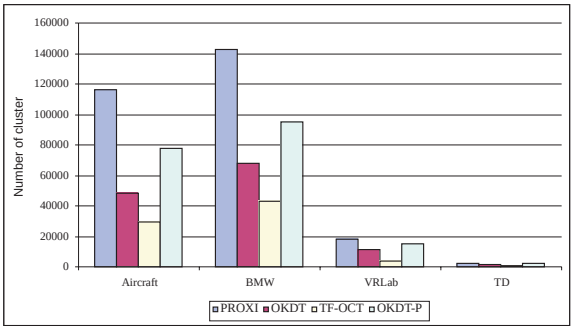


OKDT (281 s)

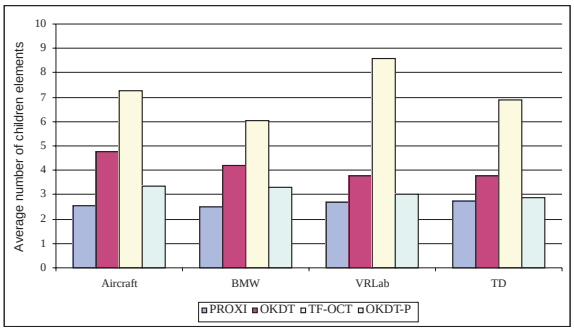


TF-OCT (166 s)

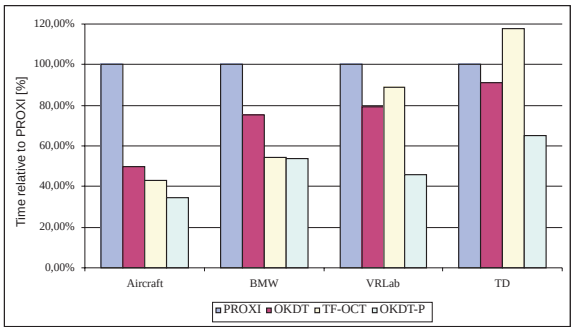
Figure 5: *Influence of the clustering method on approximate visibility calculations (see also color plate).*



Number of clusters



Avg number of children



Ray tracing time

Figure 6: *Statistics on the cluster hierarchies*

objects is possible based on connectivity and surface properties, and multi-resolution object models could be developed to provide a hierarchy of representations.

Of the clustering algorithms tested, the hierarchical bounding volumes (PROXI) approach seems to have the most predictable behavior in almost all cases. In particular, the Time-error curve is almost always monotonic and smooth. In addition, due to the nature of construction, it fits objects more tightly, which is a desirable property for clustering. However the overhead for PROXI is significant if not prohibitive in most cases: a much longer construction time (compared to all others tested), longer solution times, and in some cases a higher absolute error for very approximate simulations.

In terms of ray-casting cost, it appears that OKDT-P is the most rapid structure. Thus, if ray-casting cost is an issue (for example in interactive updates where efficiency is paramount), this may be the clustering algorithm of choice.

Finally, in terms of the quality of approximate visibility, a clear hierarchy was found with the following order PROXI (best), OKDT-P, OKDT, TF-OCT (poorest).

We hope these first conclusions will be useful to researchers and developers who wish to use clustering for hierarchical radiosity. Clearly, much remains to be done in this domain.

The error metric adopted for our tests is one of many possibilities. It is evident that different applications have different notions of error (for example in lighting design where an exact measure of energy prevails over image quality), and these different requirements will lead to different choices for clustering. These issues must be further investigated.

The initial, first-order, classification of scene "type" with respect to their behavior in the context of a clustering algorithm is an interesting avenue of research. Ideally, extensive experimentation would allow us to determine which algorithm is suitable for a given scene. This is however a very ambitious task, so even initial results would be worthy of further research.

The development of a novel clustering approach treating scenes containing many small unrelated polygons is also an interesting challenge.

To conclude, we believe that our analysis has shown the utility of clustering for many cases, identified some weaknesses of current algorithms and identified certain important properties of each algorithm with respect to their suitability for different tasks.

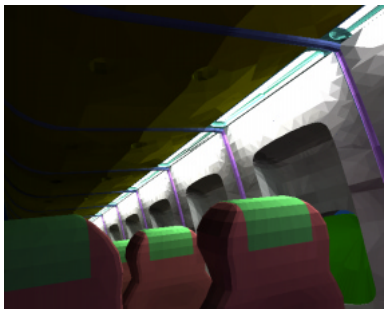
7. Acknowledgments

This work was supported by the European Union under the Esprit Long Term Research contract # 24944 "*ARCADE:*

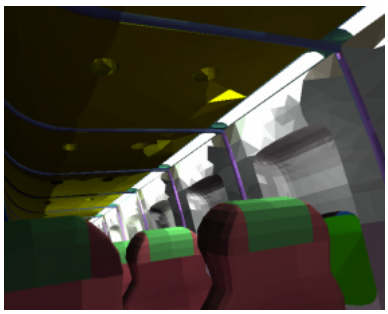
making radiosity usable" [1]. Scenes were kindly provided by Lightwork Design Ltd, Fraunhofer Institute for Computer Graphics and BMW. iMAGIS is a joint research project of CNRS/INRIA/UJF/INPG.

References

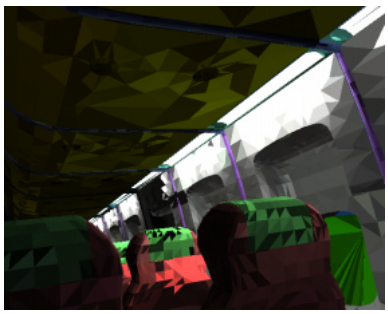
1. ARCADE: making radiosity usable. European Union. Esprit Long Term Research project #24944. `http://www-imagis.imag.fr/ARCADE`.
2. Frédéric Cazals, George Drettakis, and Claude Puech. Filtering, clustering and hierarchy construction: a new solution for ray tracing very complex environments. In F. Post and M. Göbel, editors, *Computer Graphics Forum (Proc. of Eurographics '95)*, volume 15, Maastricht, the Netherlands, September 1995.
3. Frédéric Cazals and Claude Puech. Bucket-like space partitioning data structures with applications to ray-tracing. In *Proceedings of the 13th International Annual Symposium on Computational Geometry (SCG-97)*, pages 11–20, New York, June4–6 1997. ACM Press.
4. Per Henrik Christensen, Dani Lischinski, Eric J. Stollnitz, and David H. Salesin. Clustering for Glossy Global Illumination. *ACM Transactions on Graphics*, 16(1):3–33, January 1997.
5. George Drettakis and François Sillion. Interactive update of global illumination using a line-space hierarchy. In *Computer Graphics Proceedings, Annual Conference Series:* SIGGRAPH '97 (Los Angeles, CA), pages 57–64. ACM SIGGRAPH, New York, August 1997.
6. Mark D. Fairchild. *Color Appearance Models*, chapter 3, pages 90–93. Addison Wesley, 1998.
7. Simon Gibson and Roger J. Hubbold. Efficient hierarchical refinement and clustering for radiosity in complex environments. *Computer Graphics Forum*, 15(5):297–310, December 1996.
8. Jeffrey Goldsmith and John Salmon. Automatic creation of object hierarchies for ray tracing. *IEEE Computer Graphics and Applications*, 7(5):14–20, May 1987.
9. Pat Hanrahan, David Saltzman, and Larry Aupperle. A rapid hierarchical radiosity algorithm. *Computer Graphics*, 25(4):197–206, August 1991. Proceedings SIGGRAPH '91 in Las Vegas (USA).
10. Krzysztof S. Klimaszewski and Thomas W. Sederberg. Faster ray tracing using adaptive grids. *IEEE Computer Graphics and Applications*, 17(1):42–51, January 1997.
11. François Sillion. Clustering and volume scattering for hierarchical radiosity calculations. In G. Sakas, P. Shirley, and S. Müller, editors, *Photorealistic Rendering Techniques*. Springer Verlag, 1995. Proceedings of Fifth Eurographics Workshop on Rendering (Darmstadt, Germany, June 1994).
12. François 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. (a preliminary version appeared in the fifth Eurographics workshop on rendering, Darmstadt, Germany, June 1994).
13. François Sillion and George Drettakis. Feature-based control of visibility error: A multiresolution clustering algorithm for global illumination. In *Computer Graphics Proceedings, Annual Conference Series:* SIGGRAPH '95 (Los Angeles, CA), pages 145–152. ACM SIGGRAPH, New York, August 1995.
14. Brian Smits, James Arvo, and Donald P. Greenberg. A clustering algorithm for radiosity in complex environments. In *Computer Graphics Proceedings, Annual Conference Series:* SIGGRAPH '94 (Orlando, FL), pages 435–442. ACM SIGGRAPH, New York, July 1994.
15. Andrew J. Willmott and Paul S. Heckbert. An empirical comparison of progressive and wavelet radiosity. In Julie Dorsey and Philipp Slusallek, editors, *Eurographics Rendering Workshop 1997*, pages 175–186, New York City, NY, June 1997. Eurographics, Springer Wien. ISBN 3-211-83001-4.



Reference

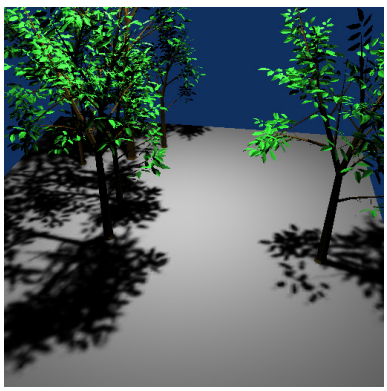


TF-OCT

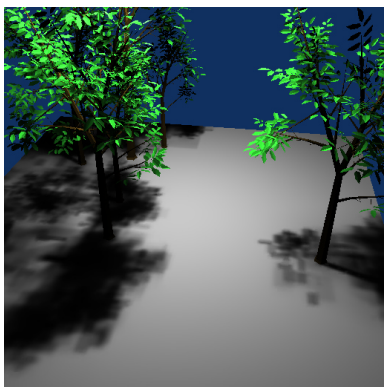


PROXI

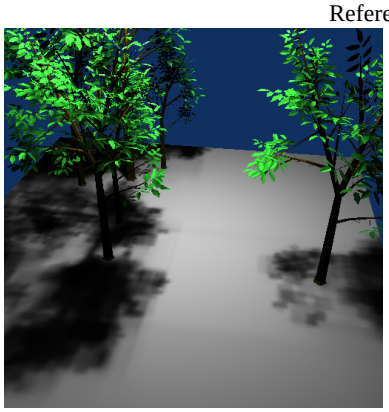
Figure 4: Example solutions exhibiting different (visual) forms of error.



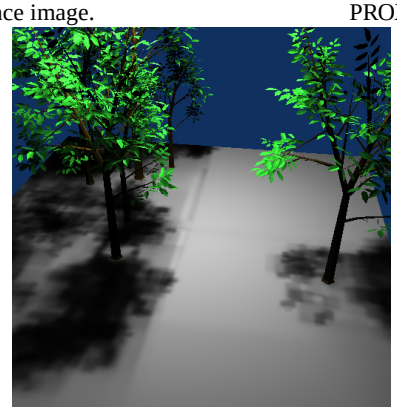
Reference image.



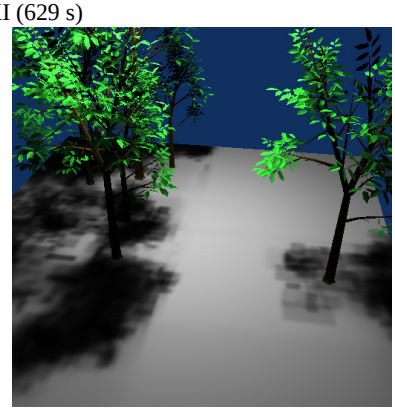
PROXI (629 s)



OKDT-P (402 s)



OKDT (281 s)



TF-OCT (166 s)

Figure 5: Influence of the clustering method on approximate visibility calculations.