Volume xxx, (*1999*) *number yyy pp. 000–000*

Efficient Glossy Global Illumination with Interactive Viewing

Marc Stamminger [1], Annette Scheel [1], Xavier Granier [2], Frederic Perez-Cazorla [3], George Drettakis [2], François Sillion [2]

[1] Max-Planck-Institute for Computer Science, Saarbrücken, Germany
[2] iMAGIS-GRAVIR/IMAG-INRIA, Grenoble, France †
[3] GGG, University of Girona, Spain

Abstract

The ability to perform interactive walkthroughs of global illumination solutions including glossy effects is a challenging open problem. In this paper we overcome certain limitations of previous approaches. We first introduce a novel, memory- and compute-efficient representation of incoming illumination, in the context of a hierarchical radiance clustering algorithm. We then represent outgoing radiance with an adaptive hierarchical basis, in a manner suitable for interactive display. Using appropriate refinement and display strategies, we achieve walkthroughs of glossy solutions at interactive rates for non-trivial scenes. In addition, our implementation has been developed to be portable and easily adaptable as an extension to existing, diffuse-only, hierarchical radiosity systems. We present results of the implementation of glossy global illumination in two independent global illumination systems.

Keywords: global illumination; glossy reflection; interactive viewing

1. Introduction

Real-world scenes contain materials with different reflective properties, varying from matte (diffuse) to shiny (glossy or specular). Global illumination research has made great advances for the treatment of diffuse environments in recent years, in particular with the advent of the Hierarchical Radiosity (HR) algorithm[8] and the subsequent introduction of clustering[21, 16]. It is now possible to compute global illumination solutions of complex diffuse environments and perform interactive walkthroughs of the result. Interactivity is achieved using the polygonal model which is appropriately subdivided into sub-polygons to capture shadows and lighting variations. Since the environments are diffuse, no updates are necessary at each frame, and the polygons are drawn as is. In contrast, scenes containing glossy surfaces cannot yet be treated in an interactive context. To generate images with glossy surfaces, ray-tracing based approaches are typically used, such as the RADIANCE system[30] or path-tracing algorithms (e.g.,[12, 26]). Some finite element approaches have been presented, but can only treat trivial scenes (e.g.,[14, 1]) or require a second, ray-casting pass to generate an image [3]. Other approaches have been proposed which are capable of interactive viewing[17, 18, 28], but they are limited in their capacity to treat non-trivial environments and reflective behaviours.

We present a novel solution which allows interactive viewing of globally illuminated glossy scenes. To achieve this goal, we use a finite element representation of outgoing radiance at surfaces or clusters. This representation is used at each frame to evaluate the radiance leaving a glossy surface and reaching the eye, permitting interactive viewing. The usage of a finite element representation for exitant light implies that the method is better suited for rough glossy surfaces, but not for highly specular or even mirror-like ones.

A novel representation of incoming radiance in the form of a structure called *Illumination Samples* is presented, which is efficient both in memory and computation time. This structure replaces an explicit (and costly) finite-element representation of incoming radiance by sets of relevant point samples.

Furthermore, we demonstrate the importance and benefits of using an adaptive hierarchical representation of outgoing radiance improving on both computation time and memory

† iMAGIS is a joint research project of CNRS/INRIA/UJF /INPG.

© The Eurographics Association 1999. Published by Blackwell Publishers, 108 Cowley Road, Oxford OX4 1JF, UK and 238 Main Street, Cambridge, MA 02142, USA.

consumption compared to previous approaches such as that of Sillion et al.[18] Our algorithm produces high quality glossy global illumination solutions which can be directly rendered for interactive walkthroughs, without the need for expensive second-pass final gather as in the work of Christensen et al.[3]

2. Previous Work

Most previous work in glossy illumination has been centered around ray-tracing. These start with distributed ray-tracing[31, 5] and the rendering equation[10] in the late eighties. A large body of research ensued focusing on Monte-Carlo stochastic algorithms. The goal of this research was to reduce the noise in the solutions, introduced by the stochastic nature of the Monte-Carlo methods (e.g.,[25, 12, 15]). Monte-Carlo algorithms that proved to be useful in other research areas were successfully transferred to the global illumination problem.[11, 27]

In parallel, several multi-pass methods have been developed which combine the advantages of ray-tracing and radiosity-style calculations;[19] others have integrated radiosity calculations in a stochastic process.[2] The RADIANCE system,[30] particle tracing [13] and photon-maps [9] are also interesting since they collect samples of illumination either on surfaces or in a separate structure, and use a ray-cast or trace to render the final image.

"Pure" finite element approaches for glossy illumination have appeared in two main flavours: three-point approaches [1, 14] and finite-element approaches using directional distributions.[18, 3] We concentrate on the last two methods in more detail, since they are closer to our new algorithm.

2.1. Wavelets and Final Gather

Christensen et al.[3] extended the wavelet methods which have been used for radiosity[32, 7, 4] to a radiance clustering algorithm. However, it still suffers from computationally expensive steps which hinder interactive viewing. Patches store a radiance distribution which is represented using a four dimensional wavelet basis accounting for spatial and directional variations. Clusters maintain a wavelet representation for an incoming as well as for an outgoing radiance distribution. The double representation of radiance consumes additional memory. Computing the transport coefficients involves evaluating a six-dimensional integral which is computationally expensive. Importance driven refinement reduces the number of interactions drastically; but the solution becomes view dependent and consequently cannot be used for interactive viewing.

Higher order wavelet bases (e.g.,[6]) were also investigated, which provide a sparser transport coefficient matrix and deliver smoother representations. However, the integrations are so complex that the authors resorted to the Haar basis with a smoothing final gather step, which is very time consuming and view dependent.

2.2. Radiance Clustering

The Radiance Clustering approach (RC) developed by Sillion et al.,[18] used spherical harmonics to store exiting radiant intensity I on the hierarchical elements of a subdivision of the original scene. An "immediate-push" algorithm is used, which, during the gather operation of light across links, "pushes" the contribution all the way to the leaves. At the leaves, radiant intensity I is stored as a spherical harmonic function; the new contribution is reflected and added into this function.

The result can be visualised directly by sampling the spherical harmonic representations of I at each frame. Direct visualisation (i.e., with no acceleration) was performed for simple scenes; since for each frame radiance is evaluated at each vertex or leaf element, frame rates are not optimal. Furthermore, spherical harmonics are a non-hierarchical representation, and the number of coefficients used is fixed in advance. As a result, there is no control over the level of detail required to represent the directionally dependent glossy illumination.

Our new algorithm provides solutions to the above problems and also reduces memory and time consumption. We start with an improved representation of incoming radiance, which avoids the memory overhead and multiple hierarchy passes of the "immediate-push" solution. In particular we introduce *Illumination Samples* which are an appropriate point sample set representation of incoming light. We then proceed with an adaptive hierarchical representation of outgoing radiance using Haar wavelets, and also present a "shooting" solution further reducing memory requirements. The resulting glossy global algorithm is well-suited to interactive viewing, and allows smooth control of the memory/quality tradeoff. This avoids the problems of non-adaptive representations which are either not sufficiently accurate or too memory-consuming. Appropriate directional refinement and simple heuristics for accelerated viewing are also introduced.

3. The Illumination Samples Algorithm

The goal of the new Illumination Samples algorithm is to extend an existing Hierarchical Clustering algorithm to also handle non-diffuse surfaces. Inter-surface light propagation is the same for diffuse and non-diffuse environments, with the difference that in a non-diffuse setup directional information about incident light must be maintained for a subsequent glossy reflection step.

As in Radiance Clustering,[18] patches and clusters are assumed to have no spatial extent as far as the representation of outgoing radiance is concerned. They store a hierarchical directional distribution for outgoing radiance which will

© The Eurographics Association 1999

be described separately in Section 4. In contrast to Christensen et al.,[3] our new algorithm does not differentiate between clusters and patches concerning the representation of exitant light.

3.1. Bounded Propagation

Our approach is based on the radiosity clustering method described by Stamminger et al.,[23, 24] which can handle flat and curved surfaces as well as clusters in a uniform manner. Bounding boxes around the objects are used to bound the set of interacting directions. With this information, bounds on the form factor and exitant radiance at the sender are computed, delivering minimum and maximum values for the received radiosity. The difference is used to decide whether to refine a link.

This bounded radiosity approach can be applied to radiance computations easily, since the propagation of light, i.e., the transformation of exitant to incident light, is independent of material properties. The only difference lies in the evaluation of bounds on the radiance of the sender, which is even easier if we have a directional distribution for the sender's exitant radiance. However, since this exitant radiance representation is only approximate, the resulting bounds are no longer conservative.

3.2. Incident Light

One way to integrate the directional information is to explicitly compute a finite element representation of it. In the work of Christensen et al.,[3] each incident light contribution computed during propagation is projected separately onto a basis for incoming light (for clusters). This is rather costly (in memory and time) and results in significant blurring of incident light, which can exhibit very strong variations. On the other hand this blurring counteracts to some extent the sharpening due to the point-representation of clusters.

An alternative is to reflect incident light contributions immediately after they have been computed,[18] while their direction of incidence is still known. The reflection responses are then projected onto finite element bases separately. This method circumvents the need to store the incident light, but the storage consumption is not reduced: two representations of exitant radiance are needed for the push/pull phase. In addition, this method is computationally expensive, because of the high number of BRDF evaluations, and the multiple hierarchy traversals involved in the immediate projection.

Our proposed solution is to combine the approaches of incident light representation and immediate reflection. We attach incident light to a receiving patch in the form of *Illumination Samples*. Light propagation is computed similarly to HR by refining links until each link represents what amounts to constant light power. Instead of simply summing the irradiance values at the receiver, an Illumination Sample with the direction to the sender and the transported irradiance is added to the receiver for each link. At the end of the propagation step, the illumination in the scene is represented as a set of point samples, distributed over the scene hierarchy (see Figure 1, left).

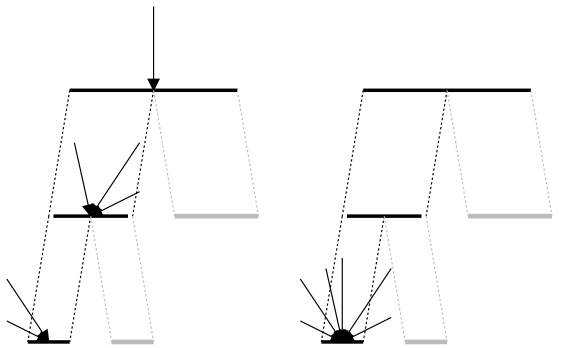


Figure 1: *Illumination Samples in the scene hierarchy. Left: after propagation computation. Right: after push.*

3.3. Push/Pull and Reflection

A `push` step as in HR is needed to create a consistent representation of the incident light at the leaves, i.e., all light received by inner nodes is propagated to the children by passing their Illumination Samples downwards. Afterwards, each leaf has a large set of Illumination Samples describing its entire incident light field (see Figure 1, right). Note that the number of hierarchy traversals is much smaller than in Radiance Clustering,[18] where each sample is pushed down separately.

After the `push` step, the incident light has to be reflected according to the object's BRDF. Because Illumination Samples correspond to Dirac impulses, the reflection is an impulse response of the BRDF, i.e., it is the BRDF with a fixed incident light direction multiplied by the irradiance of the sample. The complete reflection is the sum of the impulse responses to each Illumination Sample. Therefore the BRDF must be evaluated once for each Illumination Sample to obtain the reflected radiance in a particular direction.

Using an adaptive directional distribution described below, reflected light is projected onto an adaptive, hierarchical directional basis to obtain the new exitant light for each patch. These representations are then averaged bottom-up to obtain the distributions for inner nodes.

Due to the presence of the complete illumination information after push-pull, coherence in the incident light can be exploited for reflection computation. Consider a glossy patch being illuminated by n nearby illumination samples, all carrying approximatly the same energy (Figure 2). Each single reflection is a sharp peak, but their sum is rather smooth.

© The Eurographics Association 1999

If each incoming light transfer is reflected independently, as it is done in [18], n directional distributions are created and finally summed. As a result the directional distribution will be subdivided finely over the entire range of the reflection. Because the Illumination Sample method described above directly computes the *summed* response, it is able to detect the smooth regions of the reflection and to adapt subdivision accordingly.

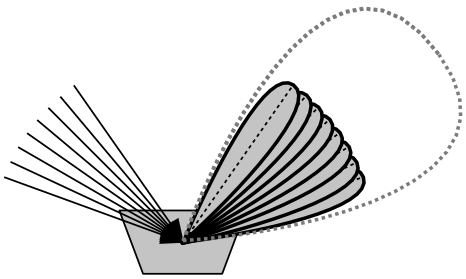


Figure 2: *Reflection of single radiance transfers on a highly glossy patch and sum of reflection distributions.*

To demonstrate this we applied the methods to the test scene shown in Figure 3. A small glossy reflector is lit by 3×7 light sources with mutual angle of 3, 15 and 30 degrees which causes the illumination of a large diffuse receiver. The reflection is computed by (a) individually reflecting each light source and summing the resulting distributions and (b) by computing a distribution of the entire reflection. For both methods the $L1$ error norm is used with the same error threshold. Table 1 shows the resulting number of basis functions, computation times and BRDF evaluations performed.

It can be seen that for all scenes the number of basis functions for the final result is significantly smaller than if the sum is projected directly. Also computation time is shorter, but the difference is not as large as one would expect. The reason can be found in the column "BRDF evaluations", where we see that their number is relatively high. This is because by projecting the sum for each exitant radiance sample all Illumination Samples are reflected, including those with neglible contribution. By projecting the responses separately, the sampling can be focused to relevant regions independently for each Illumination Sample.

3.4. Shooting

With a standard gathering iteration scheme, the number of Illumination Samples and thus the time for push/pull increases from iteration to iteration. With a shooting scheme in the spirit of [22] this can be avoided. In such a scheme, Illumination Samples are reflected once and then removed. Thus, in iteration i the light reflected exactly i times is considered only.

However, this requires the distinction of "unshot" and accumulated light. If this distinction is made for exitant light, this means that for every patch two memory-intensive directional distributions are required. With a bit more care, the distinction is made with respect to the incident light, avoiding the increase in memory consumption.

In particular, every patch gets two Illumination Sample sets, `ISSetNew` and `ISSetAccum`, as well as one directional distribtion `DD`. During iteration i (starting with $i = 0$), newly computed illumination samples are appended to `ISSetNew`. In the reflection step, `ISSetNew` is reflected and the result is stored in the directional distribution `DD`. Then the samples of `ISSetNew` are appended to `ISSetAccum` and `ISSetNew` is cleared.

This has the following consequences: in iteration i the costly directional distribution `DD` contains the light reflected exactly i times. For large i, this light is automatically represented at coarse levels of the hierarchy, so the required number of `DD`s is small. Therefore, most memory for `DD` is required in the initial iterations, and decreases continuously.

`ISSetNew` represents the incident light after exactly i reflections. It is also largest in the beginning and quickly gets smaller. Only `ISSetAccum` increases over time, but it also represents the final result, which should be stored in any case. Illumination samples are probably an efficient way of doing this.

The final solution is thus global *incident* light only. In order to obtain global exitant light, e.g., for interactive viewing (see below), `ISSetAccum` has to be reflected as a whole one final time. This is not necessary with the VISION-rendering architecture, which was used for one of the test implementations. In VISION, lighting algorithms have to compute incident light only. The last reflection step is then performed during the final ray tracing for rendering.

3.5. Discussion

Note that in our approach propagation and reflection are completely decoupled. Propagation computation does *not* consider the reflection properties of the receiver, e.g. by computing the incident light of a highly glossy patch more accurately than in the diffuse case. How this glossiness of a patch should be measured and then used to guide the accuracy of the computation, is still an open question, though. The spatial refinement of the patches is done during propagation, while the refinement level of the directional distributions is chosen during reflection. This distinction does not impose a problem on convergence, but it results in memory/computation savings.

Illumination Samples can be interpreted as Dirac-peaks from a particular direction describing incident light and are thus somewhat similar to the photons in the Photon Map approach of Jensen et al.[9] However, Photon maps are not deter-

© The Eurographics Association 1999

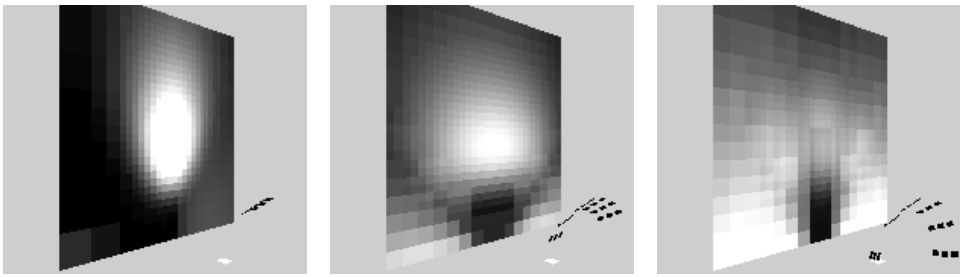


Figure 3: 3×7 *light sources illuminating a small glossy reflector, which in turn illuminates a large diffuse receiver. The angles between adjacent light sources are 3, 15 and 30 degrees.*

	reflecting the sum			summing the reflections		
light source angle	basis functions	computation time	BRDF-evaluations	basis functions	computation time	BRDF-evaluations
3	4.998	10.05s	457.968	7.140	19.83s	148.172
15	3.752	7.56s	342.000	16.390	15.89s	351.554
30	3.096	4.86s	251.790	23.574	10.39s	219.372

Table 1: *Statistics for the computation of the simple example scenes.*

ministic and their usage for lighting simulation is very different from ours.

With respect to a standard norm, with Dirac-peaks no convergent representation can be obtained. On the other hand, the Dirac-representation is only used to compute the reflection integral. From another point of view, this representation can be seen as intermediate data in a delayed numerical integration, where each Illumination Sample is a temporarily stored integration sample. So as long as the BRDF is numerically integrable, the computed reflection will converge.

The artifacts resulting from the Dirac-representation as well as the convergence of the solution due to spatial refinement are depicted in Figure 4. It shows a very simple scene of an area light and a highly glossy reflector, rendered with three decreasing error threshold values. In order to make the artifacts more visible, no smoothing is performed. It can be clearly seen how the area light is represented by 4, 16 and 64 Illumination Samples and how the reflection converges.

For a more diffuse reflector, the artifacts would be much less visible because of the larger splats. This demonstrates that light propagation towards a highly glossy patch should be subdivided finer than towards a diffuse patch. However, this requires an estimation of the glossiness of a patch and an appropriate refinement strategy. We have not pursued this issue further.

4. Adaptive Representation of Outgoing Radiance for Interactive Display

To produce the finite-element solutions suitable for interactive display, we store outgoing light in the form of directional distributions attached to surfaces or clusters. As in Radiance Clustering,[18] objects are assumed to have no spatial extent. Instead of the four dimensional radiance only the 2D *radiant intensity* distribution is stored with each object.

For the representation of directional radiant intensities, we have implemented and examined two options: First, a uniform subdivision of the direction space, where each distribution is represented by a fixed number of coefficients (*non-adaptive basis*). Second, we implemented an adaptive representation using Haar Wavelets.

Note that the use of the adaptive basis allows a comparison with RC [18]; to be complete we should have tested with spherical harmonics. Since no solution for general surface orientation currently exists, we used the “non-adaptive” basis for comparison.

The non-adaptive basis is more useful for smooth distributions, because all operations on the fixed subdivision basis are simple and fast. The adaptive Haar basis is better suited for strongly varying functions, because it can use more basis functions in the interesting regions and fewer in smooth regions. However, operations such as the evaluation of the distribution or addition of two distributions are more expensive, due to the underlying non-regular representation.

© The Eurographics Association 1999

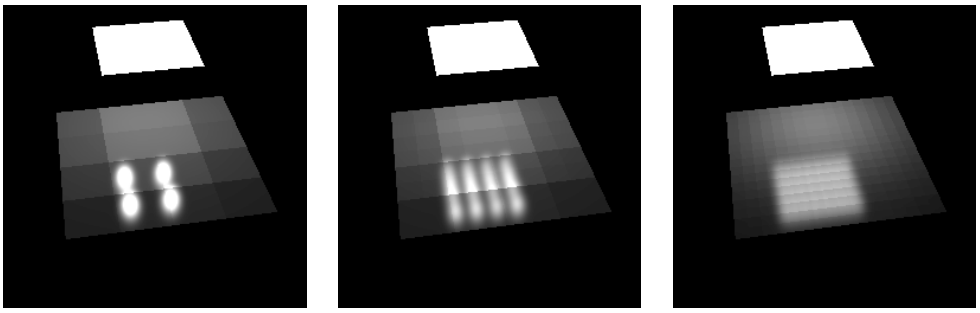


Figure 4: *Convergence of reflections of Dirac-representations. The error threshold is chosen such that the area light at the top is represented by 4, 16 and 64 Illumination Samples.*

4.1. "Non-adaptive" Representation

For a non-adaptive basis, we use a uniform subdivision of the direction space. To accomplish this task, a tetrahedron is subdivided. We thus obtain 4^{n+1} triangles if the level of subdivison is n. Since the number of vertices is lower than the number of triangles (this is $2 * 4^n + 2$), we decided to store 3 floats for RGB only at the vertices (see Figure 5).

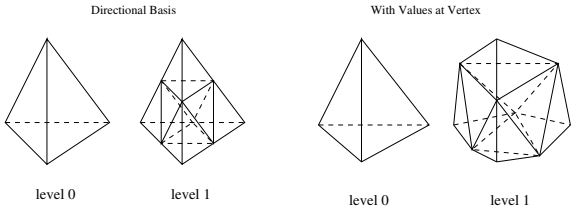


Figure 5: *Non-adaptive basis subdivision*

If some function (e.g., the reflected light of an object) is to be projected into the non-adaptive basis, the function is simply evaluated at the vertices to obtain the corresponding coefficients. As a result, this is in essence a piecewise linear representation.

4.2. Haar Representation

For the Haar representation, the domain of directions is parameterized by points on an octahedron. The vertices of the octahedron are selected to lie on the main axes, so each face corresponds to one octant of the directional domain. Simple sign considerations of a direction deliver the corresponding octahedron face.

A hierarchy of basis functions is built by assigning a first level basis function to each of the eight faces of the octahedron. These are then subdivided hierarchically in the usual manner (see Fig. 6). In order to allow for linear interpolation for later point samples, the hierarchy is always kept balanced, i.e., the subdivision levels of two neighboring triangles never differ by more than one level.

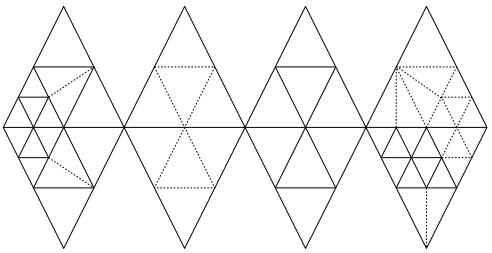


Figure 6: *Hierarchy on the octahedron.*

In order to quickly compute an adaptive representation, a top-down approach was chosen. Assume that the function to be projected is f. For each of the first eight basis functions, f is sampled at the triangle corners and at its center. If f is almost constant over the triangle, the sample values will only vary slightly. For highly varying f, one can expect a wide range of function samples. Thus the difference between minimum and maximum sample is considered. If it is too large, the four finer basis functions partitioning the domain are considered recursively.

This top-down approach runs into problems if f has a sharp peak inbetween the samples. We alleviate this problem by enforcing a minimum subdivision level in the hope that the resulting sampling is dense enough not to miss any peaks.

There are several possibilities to decide whether the difference is too large. For the algorithm described in this paper, the difference is compared with the midpoint value, i.e. a maximum *percentage* deviation ε with respect to the center value is allowed. This turned out to be beneficial for our case (especially in the context of interactive viewing, see Section 5.1); for other settings, different criteria can be used.

4.3. Comparison

To see the differences involved in using Haar or non-adaptive representations, consider the following simple ex-

© The Eurographics Association 1999

ample, which is an empty room (the geometry is taken from the RADIANCE test scene "Soda Shoppe"[29]).

The reference image was computed with a path-tracer using next-event estimation.[12] The images in Figure 7 were generated using Radiance Clustering, with the non-adaptive and Haar basis (see Section 5 for more details on rendering). The "Max Level" parameter corresponds to the maximum permitted level of subdivision. Clearly, the non-adaptive basis fails to correctly represent the highlights on the glossy floor for maximum subdivision level 3. For maximum level 4, the result is improved, but at the cost of 4 times more memory (see Table 2). In contrast, the Haar basis uses less than three times as much memory for an "equivalent" improvement in quality. However, the Haar basis also takes more time. The reason is that arithmetic operations on the regular constant subdivision are of course simpler and faster.

This example demonstrates that for highly glossy scenes, small highlights can only be captured with the adaptive basis or a very fine non-adaptive basis representation, which in turn requires large amounts of memory. More importantly, the user, who has limited memory, can only change the quality in large "quanta", and often will not be able to get a satisfactory result before running out of memory. Adaptive bases, such as Haar, alleviate this problem. However, the uniformity of the non-adaptive basis results in a smoother, more regular distribution, which becomes especially visible during interactive viewing. Note also that the Haar solution captures secondary glossy reflection from the walls to the floor, which are particularly hard for path-tracing.

Max Level	Distr	Triangles		Time	
		N/A	Haar	N/A	Haar
3	2878	782K	640K	510 s	722 s
4	2878	3127K	1829K	731 s	1125 s

Table 2: Comparison of Non-adaptive (N/A) and Haar basis for Radiance Clustering, showing the number of directional functions (Distr) used and the computation time. Max Level is the maximum level of subdivision.

5. Interactive Display

After the computation of a global illumination solution using Illumination Samples, we have a representation of outgoing radiant intensity, stored in the directional distribution function. At each frame during interactive display, we need to evaluate radiance for every glossy hierarchical leaf element in the direction of the viewpoint. This implies two requirements: (i) subdivision of the directional distributions appropriately so that a visually pleasing representation of glossy effects is produced and (ii) acceleration of the display process to avoid the cost of the evaluation of radiance at each element at every frame.

5.1. Refinement Issues for Display

Recall that we have decoupled directional subdivision, in the form of the Haar-based directional distribution functions, and the spatial subdivision, in the form of the "traditional" hierarchical radiosity element hierarchy. To interactively display the solution, we interpolate radiance in the view direction by evaluating the directional distribution on each element. If subdivision in direction space is performed arbitrarily, the difference in subdivision of the directional function between neighbouring patches may be too abrupt.

This is the case for example if we compare absolute value differences between the center and the vertices of the triangles of the directional subdivision to decide whether to subdivide. The use of relative (percentage) differences avoids this problem since we approximate the form of the function, which varies more slowly across neighbours. The artifacts due to the absolute refinement can be seen in Figure 8. In particular, note the ringing artifacts which are visible around the highlight using the absolute refiner. These artifacts are removed when using the relative (percentage) solution.

5.2. Interactive Rendering

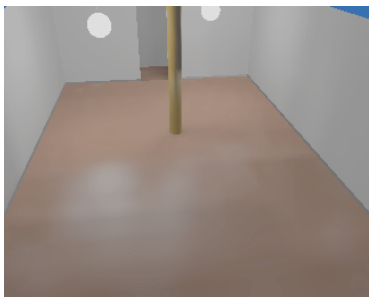
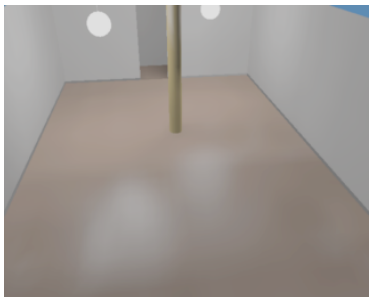
For efficient display we separate the scene into two lists, so that diffuse objects can be rendered once and redisplayed in efficient, display-list mode. The other list, of glossy reflectors, is updated appropriately at each frame and displayed in immediate mode. The accelaration achieved obviously depends on the percentage of diffuse surfaces in the scene. For the scenes tested we achieve update rates varying from a few frames per second to a few seconds per frame for more complex scenes.

In the BRIGHT rendering system, smooth interactive display of radiosity solutions is performed by storing per-vertex radiosity values. These values are updated during the push-pull phase of the solution, by extrapolating element radiosity values to the adjacent vertices.

To achieve smooth shading for glossy surfaces, we add a field to the data structure associated with vertices in the hierarchy of elements. For planar surfaces, this field is updated during push-pull in a manner slightly different to that of radiosity; i.e., for a vertex belonging to a leaf element or to an edge, the radiant intensity is summed with the radiant intensity stored at the vertex. Since radiant intensity is in Watts/sr (see [18]), at display time we divide by the area of the surrounding elements.

The special case of indexed face-sets is treated separately. Indexed face-sets are common modelling primitives, and often result from the tesselation of curved objects such as spheres or cylinders. In the BRIGHT rendering system, we pre-tesselate such objects, and represent them as an indexed face-set. The advantage of such a representation is that vertices are shared between adjacent elements. We can thus

© The Eurographics Association 1999

Max level	3	4
Non-adaptive		
Haar		