EUROGRAPHICS '98 / N. Ferreira and M. Göbel
(Guest Editors)

Volume 17, (1998), Number 3

Getting Rid of Links in Hierarchical Radiosity

M. Stamminger, H. Schirmacher, Ph. Slusallek, and H.-P. Seidel [†]

Computer Graphics Group [‡]
University of Erlangen-Nuremberg

Abstract

Hierarchical radiosity with clustering has positioned itself as one of the most efficient algorithms for computing global illumination in non-trivial environments. However, using hierarchical radiosity for complex scenes is still problematic due to the necessity of storing a large number of transport coefficients between surfaces in the form of links. In this paper, we eliminate the need for storage of links through the use of a modified shooting method for solving the radiosity equation. By distributing only unshot radiosity in each step of the iteration, the number of links decreases exponentially. Recomputing these links instead of storing them increases computation time, but reduces memory consumption dramatically. Caching may be used to reduce the time overhead. We analyze the error behavior of the new algorithm in comparison with the normal gathering approach for hierarchical radiosity. In particular, we consider the relation between the global error of a hierarchical radiosity solution and the local error threshold for each link.

1. Introduction

Global illumination has attracted much attention during the last years. Starting as a purely academic research topic in the beginning, it now has become a state-of-the-art means for generating realistic synthetic images. Radiosity has placed itself as one of the most popular illumination methods for computing indirect lighting and producing such effects as color bleeding and soft shadows.

With the quickly growing number of radiosity applications today, the size of typical scenes as well as the complexity of the corresponding illumination simulation increases drastically, resulting in large numbers of patch-to-patch links that represent the energy exchange between finite surface elements in the scene.

From among the existing radiosity algorithms, hierarchical methods have proved to be robust and easy to use. For moderately complex scenes, they are also most efficient. For complex scenes, the immense number of links requires a large amount of memory, which can be considered the major drawback of existing hierarchical algorithms[1].

A closer examination reveals that the need to store links is due to the gathering iteration scheme used for solving the radiosity equations. Since all the energy from a sender is gathered in every iteration, links are used over and over, and can never be deleted. In contrast, progressive radiosity[2] distributes only *unshot* energy in each iteration. This unshot energy decreases exponentially during the solution process.

In our new approach, we introduce a modified *shooting* iteration scheme for hierarchical radiosity, which leads to a decreased probability of reusing a once established link. This approach eliminates the need for storing all links, making the algorithm much more economical in terms of memory consumption.

Although certain links may have to be recomputed, the resulting increase in computation time may be well justified in order to avoid the very strict limitations on available memory. Furthermore, using a fixed-size cache allows us to keep around those links that have the largest possibility of being reused in later iterations and helps to minimize the performance penalty.

It is interesting to examine the error behavior of the new algorithm. In contrast to the self-correcting Gauss-Seidel scheme, it accumulates error in each iteration, which requires a detailed analysis of the two different schemes. Fortunately, the number of iterations needed to obtain a solution

[†] email: *lastname*`@informatik.uni-erlangen.de`
[‡] URL: `http://www9.informatik.uni-erlangen.de/`

of a given accuracy increases very slowly, so that for an appropriate termination criterion we can show convergence of the shooting approach.

We start the presentation with a brief review of existing radiosity algorithms in Section 2 and discuss the relative merits of the gathering and shooting approaches. The new algorithm is then described in Section 3 and its error behavior is studied in Section 4. Finally, we present statistical comparisons of the new method with standard Hierarchical Radiosity in Section 5.

2. Previous Work

The Radiosity method is described in detail in several text books[3, 4, 5]. Hence, we only state the basic facts required for our later discussion and introduce the notation used throughout the paper.

2.1. Radiosity Notation

Radiosity methods model the light transfer in a scene with purely diffuse reflectors only. The radiosity function $B(x)$, describing the radiosity at a surface point x, is the solution of the integral equation

$$B = E + \mathbf{T}B, \tag{1}$$

where $E(x)$ is the self-emission and T is the *transport operator*[6] that propagates some radiosity distribution B through the scene once and reflects it at the receiver. Equation 1 can also be written as an infinite sum

$$\begin{aligned} B &= E + \mathbf{T}E + \mathbf{T}^2E + \mathbf{T}^3E + \dots & (2) \\ &= \sum_{i=0}^{\infty} \triangle B^{(i)} \quad \text{with} \quad \triangle B^{(i)} = \mathbf{T}^i E & (3) \\ &= \lim_{n\to\infty} B^{(n)} \quad \text{with} \quad B^{(n)} = \sum_{i=0}^{n} \triangle B^{(i)}, & (4) \end{aligned}$$

which simply states that the radiosity is the sum of the emission $\triangle B^{(0)} = E$, plus the once reflected light $\triangle B^{(1)} = \mathbf{T}E$, plus the twice reflected light $\triangle B^{(2)} = \mathbf{T}^2E$ and so on.

2.2. Classical Radiosity

Classical radiosity[7] discretizes the environment into a finite set of patches and assumes that the radiosity distribution varies little over each patch. By using constant basis functions, B and E are then approximated by the average radiosity values $\hat{B}$ and $\hat{E}$ of each patch. The transport operator $\mathbf{T}$ is represented as a matrix containing transport coefficients or form factors T_{pq}, which describe the light exchange between two patches, and surface reflectances. The radiosity function is computed by solving the resulting linear system for $\hat{B}$, which is usually not done by matrix inversion, but by an iterative method like Jacobi iteration, Gauss-Seidel iteration, or Southwell relaxation[3, 4, 8].

Two main streams of algorithms have emerged to improve on the simple concept of classical radiosity: progressive radiosity[2] with several variants[9, 8, 10, 11, 12] on one side, and hierarchical approaches on the other side[13] with extensions like clustering[14, 15, 16] and wavelet radiosity[17]. The main difference between those two streams is the iteration scheme used for computing the solution $\hat{B}$.

2.3. Progressive Radiosity

Progressive radiosity or progressive refinement (PR) uses the concept of *unshot* radiosity B^u for each patch. This denotes the amount of radiosity that has not yet been propagated to the environment. Initially, B^u is set to the emission E. During an iteration step, the patch s with maximum unshot power shoots this unshot radiosity into the scene. The radiosity arriving on a patch i is added to the final solution $\hat{B}_i$ as well as to B_i^u. Finally, the unshot radiosity of the sender B_s^u is set to zero.

This iteration method turns out to be equivalent to the Southwell relaxation[8], but in the context of radiosity it is usually referred to as the *shooting* method.

Standard PR uses a predefined set of patches that is not refined during one iteration. More importantly, PR does not store the matrix elements computed in each iteration but recomputes them on the fly. This is reasonable because during the first few iterations most power is distributed from only a few bright patches, which are unlikely to be reselected until the iteration is stopped. It is only during further iterations, when the distribution of radiosity has almost reached a state of equilibrium, that each patch has a similar amount of unshot power and any patch is equally likely to be selected for shooting.

It is important to note here that the most time consuming task in radiosity is the computation of the transport coefficients $\mathbf{T}_{pq}$. This computation vastly dominates the time for solving the linear system, mostly due to visibility computations. PR benefits from the fact that a visibly faithful approximation to the solution can be obtained after only a few iterations.

2.4. Hierarchical Radiosity and Clustering

Hierarchical radiosity (HR), on the other hand, adaptively refines the surfaces to make sure that the radiosity function is well approximated. For each initial surface, a hierarchy of sub-patches is established, usually in form of a quadtree. Light is transported between two patches in a top-down manner: First a constant approximation of the light arriving at the receiver due to the sender is computed. If the *local* error of this constant approximation is estimated to be smaller than an error threshold ε, the light transport is performed on the current hierarchy level, i.e. the incoming light is evenly distributed on the receiver. If the local error is too large, sender and/or receiver are subdivided and the light exchange between the children is considered recursively. Because every surface can gather energy anywhere in its hierarchy, a push-pull step is necessary to distribute the gathered light within the hierarchy and to obtain a consistent multi-resolution representation[13].

Hierarchical radiosity algorithms with clustering (HRC)[14, 15] build a *complete* hierarchy by recursively grouping related patches and clusters, yielding a single hierarchy starting at the *root cluster*, which represents the entire scene. Radiosity can be transported between clusters and patches in a single operation. The algorithm starts by transporting light from the root cluster to itself and recursively refines this link. Clustering ensures that large sets of unimportant interactions between initial surfaces can be ignored, eliminating the initial linking phase and leading to dramatic performance gains. Clustering is considered a major improvement of hierarchical radiosity, and today nearly all HR algorithms make use of some sort of clustering. Unless noted otherwise, we will always refer to hierarchical radiosity with clustering throughout this paper.

To describe HR more formally, computing a single transport step with HR applies an approximation of the transport operator $\mathbf{T}$ to the current radiosity $\hat{B}$. We describe this approximation as $HR(\varepsilon, \hat{B}) \approx \mathbf{T}\hat{B}$. The global illumination problem is then solved by iteratively applying this algorithm to a new approximation $\hat{B}$. In every iteration step, we compute $\hat{B} \leftarrow E + HR(\varepsilon, \hat{B})$, which is equivalent to the Jacobi iteration scheme[4]. In each iteration step every patch re-gathers its complete energy from all other patches, and consequently this iteration scheme is called *gathering*. For radiosity, this iteration scheme converges quickly against the final result.

Instead of Jacobi iteration the faster converging Gauss-Seidel iteration is used[13] for Hierarchical Radiosity without clustering. In this case, an initial surface is assumed to be convex and cannot gather radiosity from itself. Consequently, a gather operation for the complete hierarchy can be performed without having to enforce consistency in the hierarchy. It is sufficient to call push-pull for the receiver once after each gathering operation. This is no longer possible for clustering with a single hierarchy, so that Jacobi iteration is used instead.

In the iteration scheme described above, $HR(\varepsilon, \hat{B})$ is computed with an increasing radiosity input $\hat{B}$. Because of that, interactions computed during one application of HR will be reused in all later iterations again. Thus they are stored in the form of *links*, avoiding the need to recompute them. The majority of links is created during the first 2–3 iterations. Afterwards, only a small number of new links is added and consequently the iteration very quickly speeds up.

2.5. Wavelet Radiosity

With wavelet radiosity[17] (WR) hierarchical radiosity has also been extended to use higher order basis functions by exploiting the features of wavelets to efficiently represent smooth functions. However, wavelet methods require many more coefficients per element, leading to an explosion in the size of links (which require transport coefficients between every pair of radiosity coefficients). Furthermore, due to the discontinuity caused by changes in visibility, the assumption of smoothness is not generally true for radiosity functions. A consequence is the poor performance of WR at shadow boundaries or near strong gradients. In these (quite common) cases, the additional approximation power of the higher order basis functions cannot avoid fine subdivision, leading to enormous memory requirements for storing links[1]. For this reason, we do not consider WR for the remainder of this paper, although the same approach can be applied.

2.6. Discussion

A major problem with HRC is the necessity to store all links because all of them are reused in each iteration. Although the number of links is in average constant per patch due to the hierarchical construction, it still imposes a considerable overhead that limits the size of scenes that can be computed with HRC. A trivial solution would be not to store links but to recompute them on the fly. Unfortunately the gathering scheme reuses links in every iteration, so computation time would increase dramatically.

A more promising approach seems to be a shooting iteration scheme together with HRC. Because the number of links for every iteration decreases exponentially, the penalty for not storing, but recomputing some of the links is much smaller than it is in the case of gathering.

In [18] a hierarchical algorithm for radiosity computations is presented, which uses a progressive radiosity style iteration approach, allowing to avoid the storage of links. However, there are some restrictions such that the algorithm is not purely hierarchical and considerations of the convergence behavior of the shooting iteration scheme are not available. In the following we want to propose a fully hierarchical clustering algorithm for radiosity computations that uses a shooting iteration scheme. The storage of links can be omitted without a dramatic efficiency loss. Using a cache storing the most costly links allows to reduce this penalty.

3. Combining Shooting and HR

In order to combine shooting and HR, we integrate the concept of *unshot* radiosity with the concept of HR. Instead of propagating the full amount of radiosity in each iteration, we only transport the exponentially decreasing unshot power.

We avoid the selection of a specific sender because in a complete hierarchy as used in HRC the object with the largest power is always the root cluster. In addition, all links carry approximately the same amount of energy in HRC, so that there are no links of higher importance to be considered first in order to obtain faster initial convergence[13].

Instead, we handle all elements in the usual recursive traversal of the hierarchy. In addition, we perform a Jacobi-style iteration by updating the values of a patch only after a full traversal of the hierarchy. Formally, this means that in every iteration step we compute an approximation $\triangle\hat{B}^{(i)}$ for the unshot radiosity after i reflections, starting with the self-emittance $\hat{E}$. In each iteration this unshot radiosity is transported once through the scene obtaining the new unshot radiosity for the next iteration. The process is stopped after n iterations, when $\triangle\hat{B}^{(n)}$ falls below some threshold. We sum all unshot radiosities to obtain the final approximation $\hat{B}$. The iteration scheme is thus:

$$\triangle\hat{B}^{(0)} = \hat{E} \tag{5}$$

$$\triangle\hat{B}^{(i+1)} = HR(\varepsilon, \triangle\hat{B}^{(i)}) \tag{6}$$

$$\hat{B}^{(n)} = \sum_{i=0}^{n} \triangle\hat{B}^{(i)}. \tag{7}$$

Although this scheme is somewhat different than traditional shooting, we will refer to it as *shooting HR*.

Looking at the iteration scheme above, one drawback of the method can be seen. In opposite to the gathering approach, it is not possible to adapt the error threshold ε manually during computation, for instance by user interaction, and then compute a finer solution reusing the coarse links.

3.1. The Shooting HR Algorithm

For shooting HR, each patch p needs to store the sum $\hat{B}_p$ of radiosity collected so far, the current unshot radiosity $\triangle\hat{B}_p$, and next iteration's unshot radiosity $\triangle\hat{B}'_p$. The unshot radiosity is initialized with the self-emission, and the algorithm stops when only the root link has been used in the last iteration.

```
SHOOTING-HR()
1  $\hat{B} \leftarrow 0$
2  $\triangle\hat{B}' \leftarrow 0, \triangle\hat{B} \leftarrow \hat{E}$
3  repeat
4        TRANSPORT-LIGHT(root, root)
5        PUSH-PULL(root)
6        $\hat{B} \leftarrow \hat{B} + \triangle\hat{B}'$
7        $\triangle\hat{B} \leftarrow \triangle\hat{B}'$
8        $\triangle\hat{B}' \leftarrow 0$
9    until number of interactions $= 1$
```

After PUSH-PULL(), which operates on the newly received radiosity $\triangle\hat{B}'$, the sum $\hat{B}$ is updated. As preparation for the next iteration, the current $\triangle\hat{B}$ values are set to those collected in the past iteration, and $\triangle\hat{B}'$ is reset to 0. The core of the algorithm is the TRANSPORT-LIGHT() function, which is in fact the normal integrated refinement and light transport algorithm used in HR with clustering:

```
TRANSPORT-LIGHT($p, q$)
1  switch
2    case ORACLE($p,q$) $= p$ :
3         for all children $c$ of $p$
4         do TRANSPORT-LIGHT($c, q$)
5    case ORACLE($p,q$) $= q$ :
6         for all children $c$ of $q$
7         do TRANSPORT-LIGHT($p, c$)
8    case else :
9         $\triangle\hat{B}'_p \leftarrow \triangle\hat{B}'_p + \hat{\mathbf{T}}_{pq}\triangle\hat{B}_q$
```

The final iteration propagates the remaining radiosity to the whole scene via the root cluster's self link only. The radiosity of the root cluster, computed by the PUSH-PULL() operation, is an area-weighted average of all radiosity values. This is the equivalent to using the ambient correction term in the progressive refinement algorithm[2].

3.2. Caching of Links

It is obvious that during a complete simulation a number of links may be used more than once, so not storing them will result in a noticeable performance loss. However, it is not too difficult to employ a caching algorithm, which holds important links in a fixed size cache.

The caching algorithm should cache those links, which are most likely to be reused in the next iteration. Because the radiosity distribution in the next iteration step is unknown, we can only assume that it will be similar to the current light distribution, i.e. mainly differ from it by a constant factor: $\triangle B^{(i+1)} \approx C\triangle B^{(i)}$, where C is about the average reflectivity of the scene. In practice, this assumption is inaccurate in the first (two or three) iterations, but it is well suited for later iterations, when the remaining unshot radiosity $\triangle B^i$ becomes more equally distributed. This fits well to the fact that in the first iterations only few links can be reused anyway (see Section 5), so caching is not important in the beginning.

A straightforward approach to caching is to define a certain threshold $\varepsilon' > \varepsilon$ and to cache all links for which the local error is higher than this caching threshold. If in the next iteration step light distribution is proportional to the previous step, these links will be most likely to be used again.

If an energy based oracle is used, the number of cached links for a given caching threshold ε' can be predicted well. Thus we can select ε' appropriately for a given cache size.

4. Error and Convergence

As we will show in this section, shooting HR in theory has a worse error behavior than gathering HR[19, 20, 21]. The reason is that the shooting iteration scheme is not self-correcting such as gathering. All computational errors are summed up for the final result, whereas in gathering every intermediate estimate $\hat{B}^{(i)}$ is transported again and thus the contained error is attenuated. The difference can also be demonstrated by considering a simple scenario: if during gathering the iterated value $\hat{B}^{(i)}$ is set to some arbitrary value, the iteration finally comes back to the correct result. If in shooting some $\triangle\hat{B}^{(i)}$ is changed manually, the error influences all following $\triangle\hat{B}$ and is never corrected during later iterations. In the following, we describe this different behavior in a formal way and show that the shooting scheme nevertheless converges against the correct solution, yet generally slower than gathering.

4.1. Error of a Single HR Light Transport

Both shooting and gathering use the hierarchical radiosity scheme to approximate one application of the light transport operator $\mathbf{T}$ to a finite element radiosity representation $\hat{B}$. We denoted such an application using the local error threshold ε and input $\hat{B}$ as $HR(\varepsilon,\hat{B}) \approx \mathbf{T}\hat{B}$. ε can be used to control the global error $\delta(\varepsilon) = ||HR(\varepsilon,\hat{B}) - \mathbf{T}\hat{B}||$ of the approximation. There have been several theoretical considerations that an upper bound on the global error δ for one single iteration exists, which converges to zero with ε [22, 4]. However, to the knowledge of the authors no estimate for the relation between global and local error has been proposed up to now. Because this relation will be needed for the following considerations we derive it here.

Relation of Global and Local Error

We consider a hierarchical radiosity algorithm that uses the amount of transported power to decide about refinement. We assume, that applying this algorithm together with a local error threshold ε to a scene creates a link between two particular patches A and B. Furthermore, we assume that A and B are of similar size, which is justified after some initial subdivision steps. By transporting light by a single link from A to B mainly two errors are introduced: Firstly, approximating the illumination using a constant basis function results in a discretization error δ_D. Secondly, form factors are usually computed numerically, resulting in an integration error δ_I.

If ε is now reduced to 1/16th, the algorithm will on average decide to subdivide both sender and receiver once and establish 16 new links between each of the four children, as shown in Figure 1. Subdividing the receiver decreases the discretization error, subdividing the sender decreases the integration error. Due to the well known linear approximation power of constant basis functions, the discretization error is halved when the illumination at the receiver is represented with four instead of only one constant basis function. Assuming a very simple one-point integration rule for the form factor, the integration error however is quartered, because now four samples in the form of four links are used to determine illumination at one of the four receiving children (see

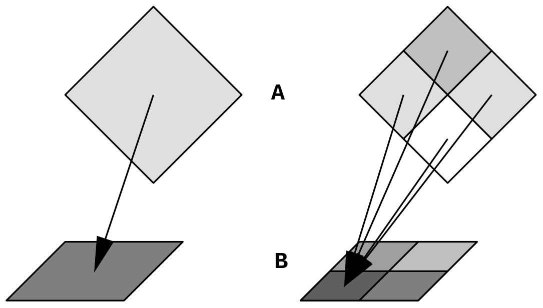


Figure 1: *Light transport between two patches A and B by one single link (left) and by 16 links after both A and B have been refined once*

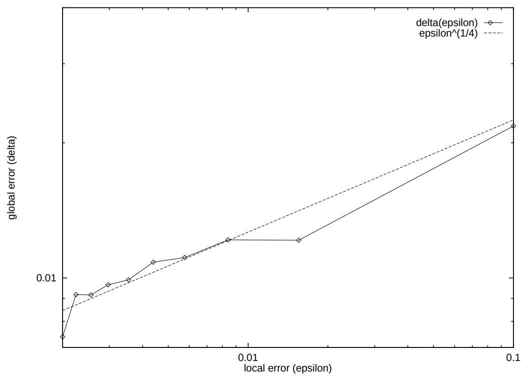


Figure 2: *Convergence of global error in HR for a test scene of 400 patches (Peter Shirley's test scene of a room with a table and four chairs). The plot shows in logarithmic scale the global error δ of the solution dependent on the local error ε allowed within one interaction. The slope of the error curve is about 1/4, indicating a relation $\delta \sim \varepsilon^{1/4}$.*

Figure 1). Thus the discretization error δ_D is proportional to $\varepsilon^{1/4}$, whereas the integration error is proportional to $\varepsilon^{1/2}$.

Because the discretization error converges worse, it dominates the global error of the solution, so we can expect the global error δ to be proportional to the local error

$$\delta \sim \varepsilon^{1/4}.$$

For testing purposes, we examined the error of a test scene of 400 polygons by comparing solutions for various ε against a reference solution obtained from a very fine hierarchical radiosity computation. The result can be seen in Figure 2. The slope of the error curve is very close to $1/4$ as expected.

Note that our considerations above are worst-case studies. For instance, if Gouraud-Shading is used to display the result, the convergence rate of the discretization error will get better. Furthermore, in most cases higher order integration is used for the form factors, especially for visibility. The impact of these improvements on the relation $\delta(\varepsilon)$ is difficult to describe mathematically. However, these improvements should all lead to a better convergence rate than $\varepsilon^{1/4}$.

4.2. Error of Gathering HR

Using the gathering iteration scheme, the following computations are performed

$$\begin{aligned}\hat{B}^{(0)} &= \hat{E} = E + R_E \\ \hat{B}^{(i+1)} &= \hat{E} + HR(\varepsilon, \hat{B}^{(i)}) \\ &= \hat{E} + \mathbf{T}\hat{B}^{(i)} + R^{(i)}_{\mathrm{HR}},\end{aligned}$$

where R_E is the error made projecting E into the finite element basis and $R^{(i)}_{\mathrm{HR}}$ is the error of the HR transport step. The iteration continues until $||\hat{B}_{i+1} - \hat{B}_i||$ is smaller than some threshold.

Through an iteration, the following errors arise

$$\begin{aligned}\hat{B}^{(0)} &= \hat{E} = E + R_E \\ \hat{B}^{(1)} &= \hat{E} + HR(\varepsilon, \hat{B}^{(0)}) \\ &= \hat{E} + \mathbf{T}\hat{E} + R^{(0)}_{\mathrm{HR}} \\ \hat{B}^{(2)} &= \hat{E} + HR(\varepsilon, \hat{B}^{(1)}) \\ &= \hat{E} + \mathbf{T}\hat{E} + \mathbf{T}^2\hat{E} + \mathbf{T}R^{(0)}_{\mathrm{HR}} + R^{(1)}_{\mathrm{HR}} \\ &\vdots \\ \hat{B}^{(n)} &= \sum_{i=0}^{n} \mathbf{T}^i \hat{E} + \sum_{i=0}^{n-1} \mathbf{T}^{n-1-i} R^{(i)}_{\mathrm{HR}}.\end{aligned}$$

For the error after n iteration steps we get

$$\hat{B}^{(n)} - B^{(n)} = \left(\sum_{i=0}^{n} \mathbf{T}^i\right) R_E + \sum_{i=0}^{n-1} \mathbf{T}^{n-1-i} R^{(i)}_{\mathrm{HR}}.$$

We know that due to energy conservation $||\mathbf{T}|| \le \tau$ for some $\tau < 1$. If we bound the discretization error R_E by δ_E and the HR errors $R^{(i)}_{\mathrm{HR}}$ by δ_{HR}, we can bound the overall error for $n \to \infty$ as

$$\begin{aligned}&\lim_{n\to\infty} ||\hat{B}^{(n)} - B^{(n)}|| \\ &\le\ ||\sum_{i=0}^{\infty} \mathbf{T}||\, ||R_E|| + ||\sum_{i=0}^{\infty} \mathbf{T} R^{(i)}_{\mathrm{HR}}|| \\ &\le\ ||(1-\mathbf{T})^{-1}||\,\delta_E + ||(1-\mathbf{T})^{-1}||\,\delta_{\mathrm{HR}} \\ &\le\ (\delta_E + \delta_{\mathrm{HR}})/(1-\tau), \qquad (8)\end{aligned}$$

which obviously converges for decreasing δ_E and δ_{HR}.

The scheme is self correcting, because the error $R^{(i)}_{\mathrm{HR}}$ is transported through the scene in the further steps and thus attenuated again. The error R_E behaves differently, because it is re-added in every step.

4.3. Error of Shooting HR

Unfortunately, the theoretical error behavior of shooting HR is not as good as for gathering HR. Tracking the transportation error throughout the iteration process we observe

$$\begin{aligned}\triangle\hat{B}^{(0)} &= \hat{E} = E + R_E \\ \triangle\hat{B}^{(1)} &= \mathbf{T}\triangle\hat{B}^{(0)} + R^{(0)}_{\mathrm{HR}} = \mathbf{T}\hat{E} + R^{(0)}_{\mathrm{HR}} \\ \triangle\hat{B}^{(2)} &= \mathbf{T}\triangle\hat{B}^{(1)} + R^{(1)}_{\mathrm{HR}} = \mathbf{T}^2\hat{E} + \mathbf{T}R^{(0)}_{\mathrm{HR}} + R^{(1)}_{\mathrm{HR}} \\ &\vdots \\ \triangle\hat{B}^{(n)} &= \mathbf{T}^n E + \mathbf{T}^n R_E + \sum_{i=0}^{n-1} \mathbf{T}^{n-1-i} R^{(i)}_{\mathrm{HR}}.\end{aligned}$$

Comparing the first n terms of the exact solution to the shooting solution after n steps we get

$$\begin{aligned}&\hat{B}^{(n)} - B^{(n)} \\ &= \sum_{i=0}^{n} (\triangle\hat{B}^{(i)} - \triangle B^{(i)}) \\ &= \sum_{i=0}^{n} \left(\sum_{j=0}^{i-1} \mathbf{T}^{i-1-j}\right) R^{(j)}_{\mathrm{HR}} + \sum_{i=0}^{n} \mathbf{T}^i R_E.\end{aligned}$$

Using the relation $||T|| \le \tau$, we obtain the error bound

$$\begin{aligned}&||B^{(n)} - \hat{B}^{(n)}|| \\ &\le \left(\sum_{i=0}^{n} ||(1-\mathbf{T})^{-1}||\, ||R^{(i)}_{\mathrm{HR}}||\right) + ||(1-\mathbf{T})^{-1}||\, ||R_E|| \\ &\le\ ((n+1)\delta_{\mathrm{HR}} + \delta_E)/(1-\tau). \qquad (9)\end{aligned}$$

The problem of shooting becomes visible in this formula. The error of the result increases with the number of iterations n. The reason is that every iteration computes a part of the result, so if every iteration produces some error, more iterations also produce more error. Note, that the assumption that every iteration produces an error of about the same magnitude is very pessimistic, because in each iteration also the amount of transported light decreases. In practice the error will get smaller for later iterations. For gathering however, every iteration computes a new value for B using the previous value only as a starting value, thus errors in B are corrected during later iterations.

Fortunately, it is not necessary to use an arbitrarily large n. As described before, we stop iteration as soon as only a single interaction via the root link is used. This number of iterations can be bounded a priori. If we denote the maximum emission $\lceil E \rceil$, we know that $||\triangle B^{(n)}|| < \tau^n \lceil E \rceil$. An upper bound for the iteration $n(\varepsilon)$ in which $||\triangle B^{(n)}||$ becomes smaller than ε is thus

$$n(\varepsilon) = \left\lceil \frac{\log(\varepsilon/E)}{\log\tau} \right\rceil, \qquad (10)$$

which means that n grows very slowly when ε decreases. If

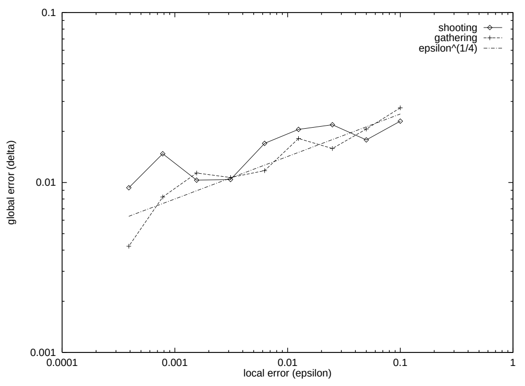


Figure 3: *Correlation of global error δ and local error ε for both gathering and shooting for the Shirley table scene. For comparison to the estimate derived in Section 4.1 a curve for* $\delta \sim \varepsilon^{1/4}$ *is also given.*

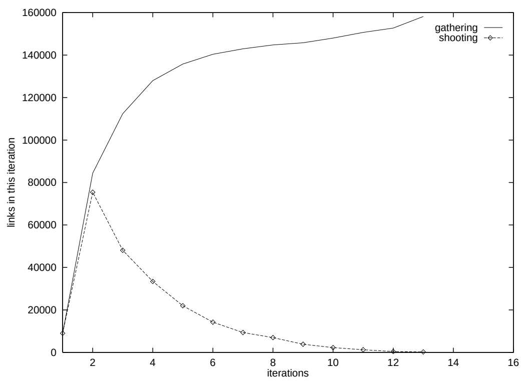


Figure 4: *Links required for light transport as a function of the number of iterations for* $\varepsilon = 3 \cdot 10^{-3}$*. The number of links increases for gathering HR and falls off exponentially after the second iteration for shooting HR.*

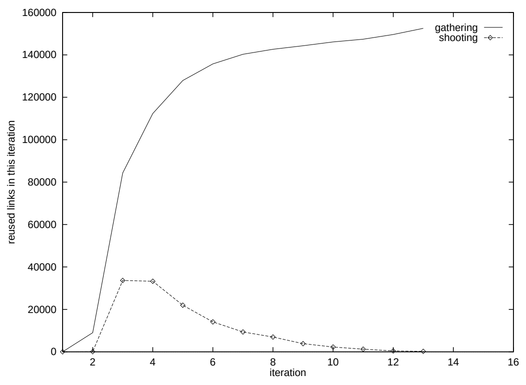


Figure 5: *Number of potentially reused links. While for gathering nearly all links are used over and over, shooting only reuses a small fraction of its links in the beginning and a larger fraction of the small absolute number later on.*

we assume that $\delta = C\varepsilon^{1/4}$ and put Equation 10 into Equation 9, the resulting error converges to zero, because $\delta_{\mathrm{HR}}(\varepsilon)$ converges faster to zero than $n(\varepsilon)$ diverges.

5. Results

We have implemented the shooting HR algorithm within the physically-based rendering architecture VISION[23], mainly by adapting a REFINE-LINK() method in the radiosity subsystem to a new TRANSPORT-LIGHT() function. Our hierarchical radiosity algorithm uses a truly error-bounding refiner extended to handle clusters and arbitrary types of (curved) objects[24, 16].

In the following, we show link statistics for the shooting algorithm in order to demonstrate the behavior of shooting HR compared to gathering HR. We also discuss error statistics by comparing the shooting and gathering results to a reference solution.

5.1. Error Analysis

We tested the algorithm on one of the test scenes of Peter Shirley (a room with a table and four chairs with 400 surfaces) using a sequence of decreasing local error thresholds ε. Figure 3 shows the resulting global error δ of both gathering and shooting HR plotted against the local error threshold ε and compared to a curve describing $\delta \sim \varepsilon^{1/4}$. Gathering HR yields more accurate results for the same ε, but clearly shooting HR is not very far off. Both curves are in accordance with the relationship discussed in Section 4.1.

Looking at the resulting images it is interesting to note that there is not visually noticeable difference in the solution. The reason is that for shooting the main error arises in the later steps of the iteration and not in the first two or three iterations, during which the visually important, non-smooth illumination is computed.

5.2. Links

In order to compare gathering and shooting HR, we present link statistics for both of them in Figures 4 and 5 using two equivalent local error thresholds which lead to approximately the same global error δ.

Figure 4 shows the number of links required during each iteration for shooting and gathering. It can clearly be seen that after the first two iterations the number of links required for shooting falls off exponentially, whereas for gathering it converges against some maximum. Using the gathering scheme without storing the links would thus result in an enormous increase in computation time, whereas for shooting this seems reasonable.

It is also very interesting to examine the number of potentially reusable links. Figure 5 shows the number of links required in each iteration that have already been computed

iteration	links required	cache size 100,000 cached	cache size 100,000 reused		cache size 1,000,000 cached	cache size 1,000,000 reused	
1	420,435	0	0		0	0	
2	1,415,249	1,258	1,080	(0%)	137,018	1,080	(0%)
3	1,002,709	57,806	49,804	(5%)	972,762	495,626	(49%)
4	758,932	82,643	67,170	(9%)	1,179,903	592,602	(78%)
5	513,757	95,942	66,989	(13%)	1,194,717	448,420	(87%)
6	352,941	105,720	60,474	(17%)	1,197,048	326,134	(92%)
7	234,509	111,955	51,461	(22%)	1,198,976	222,428	(95%)
8	158,517	117,431	43,627	(28%)	1,199,235	152,608	(96%)

Table 1: *Link cache statistics for an example scene propagating light via 4.8 million single interactions using cache sizes of 100,000 and 1,000,000*

cache size	computation time	computed links	visibility tests
100,000	3,850s	3,851,997	9,705,844
1,000,000	3,068s	2,621,388	7,458,368
∞	2,859s	2,189,660	7,123,937

Table 2: *Computation times for example scene using cache sizes of 100,000, 1,000,000, and unlimited cache size*

previously and can thus be reused if cached. For gathering, almost all links can be reused.

For shooting, however, in the second iteration almost no links can be reused, because all links from the first iteration emanate from the light sources. For later iterations, these links are not relevant anymore. Even for the third iteration, the number of reusable links is small. From the forth iteration on, most links could be reused, but this number of reusable links is only one fifth of the total number of links to be stored for gathering.

5.3. Link Cache

In order to test the link caching mechanism, we computed a rather fine solution of an office scene consisting of almost 5,000 initial patches, which were subdivided into about 40,000 patches during 8 iterations. 2.2 million links are used during computation, light is transported in 4.8 million interactions, and each of the links is reused about 1.2 times on average. Simply storing no links at all thus about doubles computation time. The result can be seen in Figure 6.

Table 1 shows for each iteration the number of links in the cache from the previous step, the number of links required for the current step, and the number and percentage of links that could be reused from the cache for a preselected cache size of 100,000 and 1,000,000 links.

Table 2 shows the computation times for cache sizes of 100,000, 1,000,000 and for an arbitrary large link cache, as well as the number of computed links and the number of visibility tests. It is interesting to note that the computation time does not increase proportionally with the number of links that are computed. The reason can be seen in the number of visibility tests: We choose the number of visibility tests for an interaction depending on the amount of transported light. Interactions with high energy are thus more costly than low energy interactions, so more expensive interactions are cached, whereas the cheaper low energy links are recomputed.

5.4. Large Scene

As last example we computed an accurate radiosity solution for a fairly complex scene of more than 75,000 initial patches, which were subdivided into 300,000 final patches during computation (Figure 7). Light was transported in 5 iterations via a total of 35,000,000 links. The overall computation took 20 hours on a SGI Onyx with MIPS R10k. A link cache for 1,000,000 links was used to speed up computation. Assuming a size of 16 bytes per link, we used an extremely small link cache of only 16 MB. Storing all links would have required a total of 560 MB only for the links, not accounting for the storage required for the scene database itself.

6. Conclusions

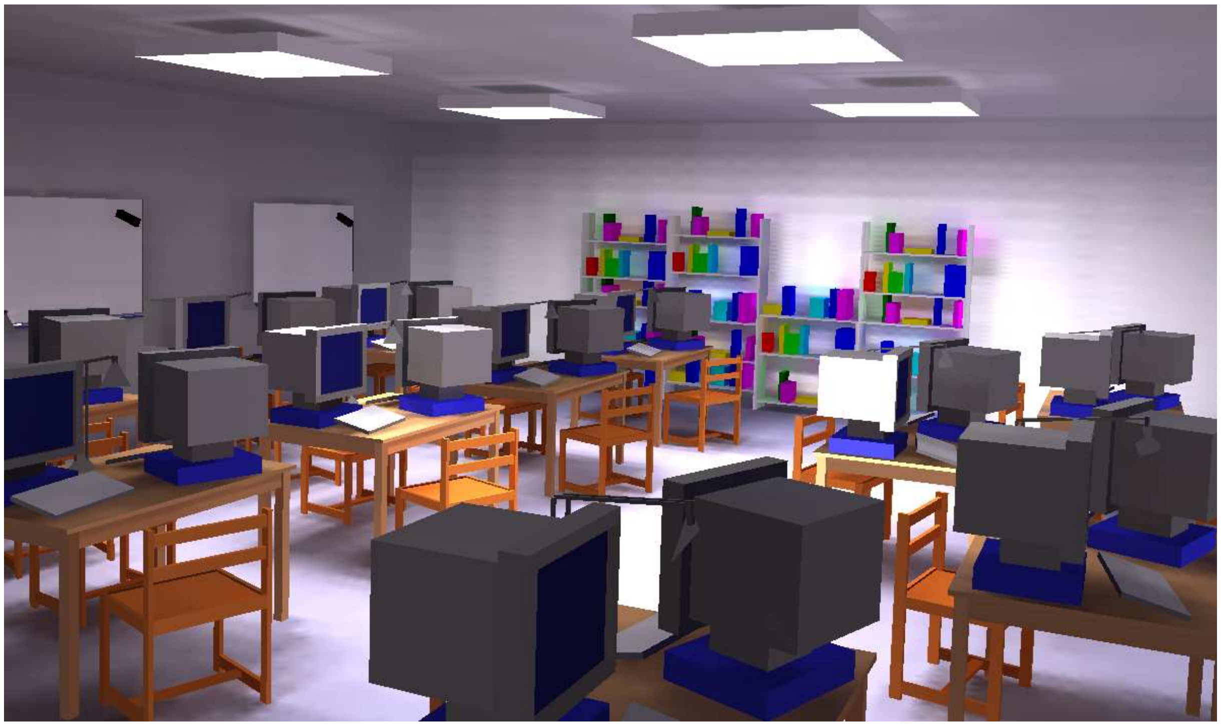
In this paper we showed how a shooting iteration scheme can be used together with Hierarchical Radiosity. Due to the usage of shooting instead of gathering the reuse of links is small, so the storage of links can be avoided. Memory requirement, which is a major drawback of current Hierarchical Radiosity algorithms, is kept low with this approach. The savings in memory requirements are bought by a modest increase in computation time and error. Nevertheless, it turned out that using a fixed size cache for costly links can diminish the increased computation time and that the increase in error in practice is small and visually not noticeable.

We also demonstrated theoretically that the new method converges to the correct solution, yet slower than the standard gathering approach. For that purpose we derived an asymptotic relation between global and local error of Hierarchical Radiosity.

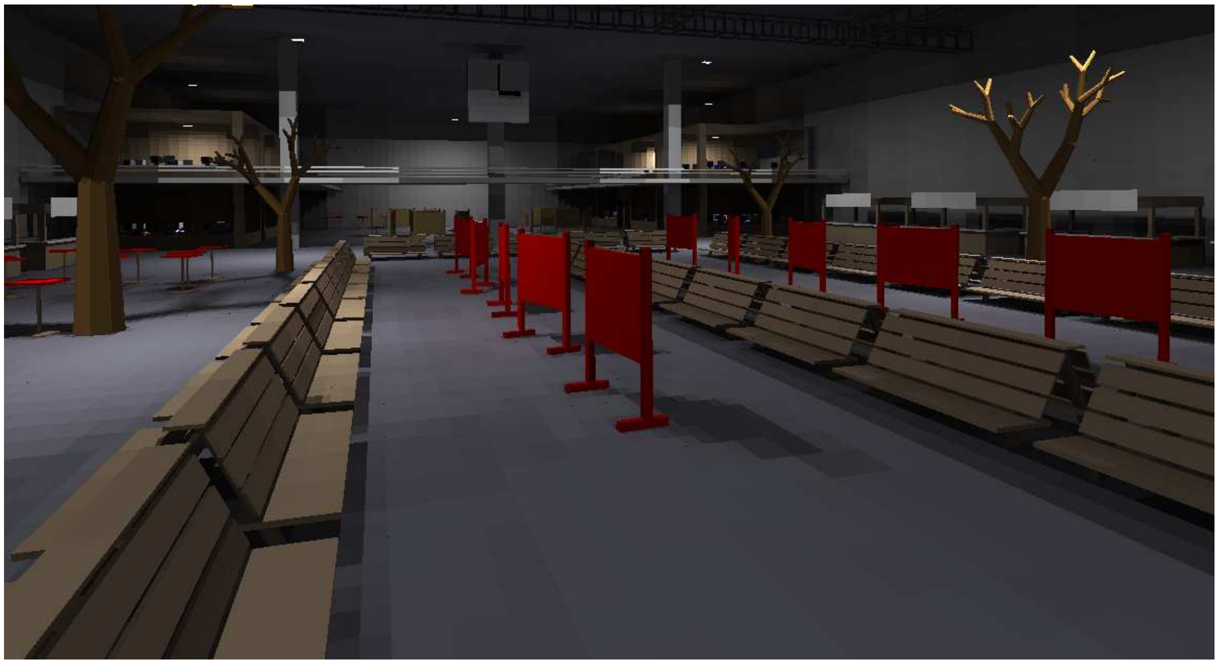
Acknowledgments The authors would like to thank Leif Kobbelt for fruitful discussions regarding this topic and Annette Scheel for her help to finish this paper.

References

1. A. Willmott and P. Heckbert, “An empirical comparison of radiosity algorithms”, Tech. Rep. CMU-CS-97-115, Carnegie Mellon University, Pittsburgh, (1997).
2. M. Cohen, S. Chen, J. Wallace, and D. Greenberg, “A progressive refinement approach to fast radiosity image generation”, in *Computer Graphics (SIGGRAPH ’88 Proceedings)*, vol. 22, pp. 75–84, (1988).
3. M. Cohen and J. Wallace, *Radiosity and Realistic Image Synthesis*. San Diego, CA: Academic Press Professional, (1993).
4. F. Sillion and C. Puech, *Radiosity and Global Illumination*. San Francisco: Morgan Kaufmann, (1994).
5. A. Glassner, *Principles of Digital Image Synthesis*. Morgan Kaufmann Publishers Inc., (1995).
6. J. Arvo, “The role of functional analysis in global illumination”, in *Rendering Techniques ’95 (Proceedings Sixth Eurographics Workshop on Rendering)*, (Dublin), pp. 115–126, Springer, (1995).
7. M. F. Cohen, D. P. Greenberg, D. S. Immel, and P. J. Brock, “An efficient radiosity approach for realistic image synthesis”, *IEEE Computer Graphics and Applications*, **6**(3), pp. 25–35 (1986).
8. S. Gortler, M. Cohen, and P. Slusallek, “Radiosity and relaxation methods”, *IEEE Computer Graphics & Applications*, **14**(6), pp. 48–58 (1994).
9. M. Feda and W. Purgathofer, “Accelerating radiosity by overshooting”, in *Third Eurographics Workshop on Rendering*, (Bristol, UK), pp. 21–32, (1992).
10. G. Greiner, W. Heidrich, and P. Slusallek, “Blockwise refinement – a new method for solving the radiosity problem”, in *Proceedings Fourth Eurographics Workshop on Rendering ’93*, pp. 233–246, Springer, (1993).
11. L. Neumann, W. Purgathofer, R. F. Tobler, A. Neumann, P. Eliás, M. Feda, and X. Pueyo, “The stochastic ray method for radiosity”, in *Rendering Techniques ’95 (Proceedings Sixth Eurographics Workshop on Rendering)*, (Dublin), pp. 206–218, Springer, (1995).
12. M. Sbert, F. Pérez, and X. Pueyo, “Global monte carlo. a progressive solution”, in *Rendering Techniques ’95 (Proceedings Sixth Eurographics Workshop on Rendering)*, (Dublin), pp. 231–239, Springer, (1995).
13. P. Hanrahan, D. Salzman, and L. Aupperle, “A rapid hierarchical radiosity algorithm”, in *Computer Graphics (SIGGRAPH ’91 Proceedings)*, vol. 25, pp. 197–206, (1991).
14. F. Sillion, “A unified hierarchical algorithm for global illumination with scattering volumes and object clusters”, *IEEE Transactions on Visualization and Computer Graphics*, **1**(3), pp. 240–254 (1995).
15. B. Smits, J. Arvo, and D. Greenberg, “A clustering algorithm for radiosity in complex environments”, in *Computer Graphics (SIGGRAPH ’94 Proceedings)*, pp. 435–442, (1994).
16. M. Stamminger, P. Slusallek, and H.-P. Seidel, “Bounded radiosity – illumination on general surfaces and clusters”, *Computer Graphics Forum (EUROGRAPHICS ’97 Proceedings)*, **16**(3), (1997).
17. S. Gortler, P. Schröder, M. Cohen, and P. Hanrahan, “Wavelet radiosity”, in *Computer Graphics (SIGGRAPH ’93 Proceedings)*, vol. 27, pp. 221–230, (1993).
18. K. Myszkowski and T. L. Kunii, “An Efficient Cluster-based Hierarchical Progressive Radiosity Algorithm”, in *Third International Computer Science Conference: Image Analysis Applications and Computer Graphics (ICSC ’95)*, vol. 1024 of *Lecture Notes in Computer Graphics*, pp. 292–303, Springer, (1995).
19. D. Lischinski, B. Smits, and D. Greenberg, “Bounds and error estimates for radiosity”, in *Computer Graphics (SIGGRAPH ’94 Proceedings)*, pp. 67–74, (1994).
20. J. Arvo, K. Torrance, and B. Smits, “A framework for the analysis of error in global illumination algorithms”, in *Computer Graphics (SIGGRAPH ’94 Proceedings)*, pp. 75–84, (1994).
21. P. Bekaert and Y. D. Willems, “Error control for radiosity”, in *Rendering Techniques ’96 (Proceedings Seventh Eurographics Workshop on Rendering)* (X. Pueyo and P. Schröder, eds.), (Porto), pp. 153–164, Springer, (1996).
22. P. Schröder, *Wavelet Algorithms for Illumination Computations*. PhD thesis, Princeton University, (1994).
23. P. Slusallek and H.-P. Seidel, “Vision: An architecture for global illumination calculations”, *IEEE Transactions on Visualization and Computer Graphics*, **1**(1), pp. 77–96 (1995).
24. M. Stamminger, P. Slusallek, and H.-P. Seidel, “Bounded clustering – finding good bounds on clustered light transport”, Tech. Rep. IMMD9-2, University of Erlangen-Nuremberg, (1997).



[Stamminger et al.] Figure 6: *Office scene consisting of 5,000 initial patches subdivided during the computation into 40,000 patches.*



[Stamminger et al.] Figure 7: *Computation result of a station scene. The initial complexity of the scene are 75,000 patches, the result contains 300,000 patches. The scene was computed with a link cache size of 1,000,000 links.*