Thirteenth Eurographics Workshop on Rendering (2002)
P. Debevec and S. Gibson (Editors)

Interactive Global Illumination Using Selective Photon Tracing

Kirill Dmitriev, Stefan Brabec, Karol Myszkowski and Hans-Peter Seidel

Max-Planck-Institut für Informatik, Saarbrücken, Germany

Abstract

We present a method for interactive global illumination computation which is embedded in the framework of Quasi-Monte Carlo photon tracing and density estimation techniques. The method exploits temporal coherence of illumination by tracing photons selectively to the scene regions that require illumination update. Such regions are identified with a high probability by a small number of the pilot photons. Based on the pilot photons which require updating, the remaining photons with similar paths in the scene can be found immediately. This becomes possible due to the periodicity property inherent to the multi-dimensional Halton sequence, which is used to generate photons. If invalid photons cannot all be updated during a single frame, frames are progressively refined in subsequent cycles. The order in which the photons are updated is decided by inexpensive energy- and perception-based criteria whose goal is to minimize the perceivability of outdated illumination. The method buckets all photons on-the-fly in mesh elements and does not require any data structures in the temporal domain, which makes it suitable for interactive rendering of complex scenes. Since mesh-based reconstruction of lighting patterns with high spatial frequencies is inefficient, we use a hybrid approach in which direct illumination and resulting shadows are rendered using graphics hardware.

Categories and Subject Descriptors (according to ACM CCS): I.3.7 [Computer Graphics]: Three-Dimensional Graphics and RealismAnimation;

1. Introduction

Synthesis of realistic images which predict the appearance of the real world has many important engineering applications including product design, architecture and interior design, and illumination engineering. One of the major components of such predictive image synthesis is global illumination. Even for less rigorous applications such as special effects, film production, and computer games global illumination greatly improves the appearance and believability of rendered images.

The vast majority of existing global illumination solutions are designed for off-line computations and for static scenes. The response times required by such solutions to update illumination information in dynamically changing environments are prohibitively large even for minor changes in the scene, because the whole computation is usually repeated from scratch. Some successful attempts have been made to upgrade those algorithms to exploit the temporal coherence and reuse valid lighting information computed previously. While interactivity is often achieved for some simple scene changes, the storage costs involved in the data structure extensions into the temporal domain are intractable for practical applications dealing with complex scenes.

In this paper we propose a novel global illumination technique designed specifically for interactive applications. Our algorithm exploits a periodicity property[29] of the multi-dimensional Halton sequence, which provides immediate information on the similarity of photon paths in the scene. This enables selective photon tracing into the scene regions whose illumination must be updated due to the user interaction. To improve the progressivity of such update energy- and perception-based criteria are introduced whose main goal is to minimize the image artifacts as perceived by the observer. The unique feature of our selective photon tracing is that while the temporal coherence in lighting computations is strongly exploited, no additional data structures in-

volving photon paths in the dynamically changing scene are required. Moreover, since the photon hit points are bucketed on-the-fly directly into the corresponding elements of dense mesh the photon storage is completely avoided. In practice this means that if the mesh data structures required for efficient ray (photon) tracing fit into the computer memory the global illumination computation can be performed using our method. This makes our technique suitable for interactive global illumination even for complex scenes. In terms of hardware requirement a simple PC-class computer equipped with a graphics board offers sufficient performance for our technique, and expensive multiprocessor computers or computer clusters are not required.

Our rendering technique is optimized for interactivity and some of its design decisions clearly trade off image quality for computation speed. The most severe limitation of our technique is the use of simple photon bucketing for the mesh-based reconstruction of indirect illumination (direct illumination and resulting shadows are rendered using graphics hardware). Obviously, by storing photon hit points and using more advanced density estimation techniques[15, 26, 41, 18] less bias could be obtained but then the rendering interactivity would be compromised.

The paper is organized as follows. Section 2 reviews the previous approaches to global illumination in dynamic environments. In Section 3 we introduce our new framework. We discuss the periodicity property of the Halton sequence which is the foundation for our selective photon tracing in Section 4. The algorithm of interactive global illumination computation is explained in Section 5. In Section 6 we provide some implementation details of our interactive system. We discuss the results obtained by our techniques in Section 7. Finally, we conclude the paper and propose some future directions for this research.

2. Previous Work

The problem of global illumination solutions for dynamic environments has attracted significant attention of researchers, resulting in various solutions which can roughly be categorized as off-line or interactive. In the following section we briefly overview representative examples of off-line algorithms. Then we focus on interactive algorithms, which are more relevant for this paper.

2.1. Off-Line Techniques

Off-line global illumination algorithms are used in the production of high quality animations in which the accuracy of lighting simulation cannot be compromised. Many existing solutions resulted from extending hierarchical radiosity algorithms into the time domain, however, they involve huge storage costs which makes them impractical for complex scenes[7, 24]. In Global Monte Carlo Radiosity[2] the temporal coherence of costly visibility computations is efficiently and conservatively exploited, but then the radiosity solution is performed independently for each frame and all radiosity solutions are stored simultaneously in the memory. In the range-image framework[30], direct and indirect lighting are independently sampled in time for selected keyframes. The temporal frequency of sampling is adaptive to the changes in scene illumination. Interpolation is performed between the obtained solutions to derive lighting for inbetween frames, but some important lighting events between keyframes can be overlooked. To avoid this problem, sparse sampling of the scene illumination can be performed for all animation frames[27]. However, to reconstruct lighting for a given frame Myszkowski et al.[27] utilize information not only from preceding but also from the following frames, which are not available in the interactive scenario.

In general, it is difficult to adapt a typical off-line algorithm for interactive applications. All discussed algorithms require that the complete information on all animated scene elements is known in advance. Also, since the main goal of the off-line algorithms is to reduce the overall computation time required for rendering of the whole animation, the differences in the rendering time of particular frames are perfectly acceptable. Thus, the constant frame rate usually cannot be kept, which is not acceptable for an interactive scenario.

2.2. Interactive Techniques

A number of techniques handling global illumination computation for dynamic environments at interactive speeds have been presented. The techniques have been designed to trade the image quality for the response speed, and their main objective is to provide a fast response for frame-to-frame changes in the environment. Early solutions[4, 11, 25] were embedded into the progressive radiosity framework and relied on shooting the corrective energy (possibly negative) to the scene regions affected by the environment changes.

Much better performance was obtained for more recently introduced techniques that are based on hierarchical radiosity[31, 8, 33]. Those algorithms introduce mechanisms for controlling the frame rate and efficiently identifying which part of the scene is modified. However, the hierarchical radiosity framework poorly supports the light transfer between glossy and specular surfaces. This drawback can be eliminated by adding photon tracing[12] atop of the line-space hierarchy, used for diffuse surfaces[8]. The photons traversing changed scene regions are identified using a dynamic spatial data structure and are selectively updated. But still, the memory requirements in all those hierarchical approaches are extremely high due to the storage of the link structure.

Keller introduced the instant radiosity technique[19], which, although originally proposed for efficient rendering of static scenes, can easily be extended to handle dynamic environ-

ments. The technique utilizes the quasi-random walk of photons based on Quasi-Monte Carlo integration. Then each photon hit point is considered as a point light source which is processed by graphics hardware. The final image is obtained through the view-dependent accumulation of partial images generated for each resulting light source. Keller uses the Halton sequence to generate the photon hit points. A drawback here is that Keller updates the photon paths merely based on their age criterion, ignoring whether they became invalid due to the changes in the scene or not.

The algorithms proposed by Pueyo et al.[31] and Keller[19] are view-dependent which precludes interaction involving changes of camera parameters. This deficiency is eliminated by view-independent algorithms[8, 33], however, the whole scene illumination is always updated with equal priority, without taking into account the current camera parameters and other "importance" factors. If a complete update of the scene illumination cannot be performed during a single frame refresh cycle, then what becomes important is the order in which the progressive refinement in various scene regions must be performed for the subsequent frames. The main goal of such an ordering of computation is to minimize the perceivable artifacts at the intermediate stages of a global illumination update, and this problem has basically not been addressed in existing solutions. In this paper we present a view-independent approach, in which the priority of update is based on the current view and the perceivability of illumination artifacts.

Recently, various image caching schemes have been proposed, which allow for rendering of non-diffuse surfaces for changing camera positions at interactive speeds [40, 35, 36]. Our approach could benefit significantly from such solutions which could be incorporated into our interactive global illumination system. In this paper we focus on global illumination for diffuse environments due to the limitations of illumination storage in the mesh. However, during our photon tracing, arbitrary surface reflectance functions are properly processed.

3. Overview

As a framework for global illumination computation, we chose the Density Estimation Photon Tracing (DEPT) algorithm[37]. The DEPT is similar to other solutions in which photons are traced from light sources towards surfaces in the scene, and the lighting energy carried by every photon is deposited at the hit point locations on those surfaces[15, 41]. We extend the DEPT for handling dynamic environments. Also, we replace stochastic pseudo-random sequences, which are used in the DEPT to generate photon paths, by deterministic quasi-random sequences.

In our algorithm we consider the same big pool Z of photons (which are traced at the initialization stage) for the whole interactive session. We update the photons computed for previous frames, which became invalid for the current frame due to interactive changes in the scene. The main problem is to identify each invalid photon in Z and to replace it by the corresponding photon, which is traced for the current scene configuration. Our photon tracing is based on Quasi-Monte Carlo sampling with the multi-dimensional Halton sequence. We show how the periodicity property[29] of this sequence, allows for immediate identification of photons that have similar paths in the scene. Thus, for any invalid photon, it is easy to identify indices of all similar photons in Z that are likely to be invalid as well. Since the Halton sequence can be used to generate any photon path based on its index, the storage costs of photon data can be eliminated.

Our algorithm of photon updating is iterative. The user decides upon the iteration duration, which imposes a limit on the number of photons that can be traced per iteration. The photon update iteration consists of two stages. In the first stage a subset of photons from Z are traced from the light sources to the whole environment in order to identify invalid photons. In the second stage, based on the periodicity property of the Halton sequences, those invalid photons are used to generate similar photons in Z, which are likely to traverse modified scenes regions. Since it might be impossible to update all those potentially invalid photons in a single iteration, the photon update in various scene regions is ordered by inexpensive energy- and perception-based criteria according to the importance of outdated lighting artifacts as perceived by the user. Our algorithm is conservative in the sense that finally, after N_g iterations since the last change in the environment, all photons in Z are updated. In practice, when changes in the environment are local, a small number of iterations $r \ll N_g$ is required to remove all perceivable problems with the outdated illumination. The constant frame refresh rate, which is required by many practical applications, is achieved by letting the user control the iteration duration and the number of such iterations per frame.

4. Quasi-Monte Carlo Sampling

The global illumination problem involves solving high dimensional integrals. An efficient way to solve such integrals is to apply Monte Carlo techniques, which have an advantage over other integration methods because their convergence speed does not depend on the dimension of the integral.

Monte Carlo (MC) techniques are traditionally based on pseudo-random sequences. However, it was proven that for functions with bounded variation, faster convergence can be achieved using Quasi-Monte Carlo (QMC) techniques based on quasi-random sequences[17]. Although integrands usually have unbounded variation in global illumination tasks, numerical evidence has been presented that the QMC techniques still have advantage over the MC techniques for real-world scenes[16].

In the following section we examine basic properties of

the Halton sequence, which is an example for the quasi-random sequence that proved to be useful in photon tracing[19] and stochastic radiosity[28] applications.

4.1. The Halton Sequence

The Halton sequence generation is based on the radical inverse function, applied to an integer i. The radical inverse $\Phi_b(i)$ is the number obtained by expressing i in the base b, then reversing the order of the resulting digit sequence and placing the floating point at the sequence beginning[13]:

$$\Phi_b(i) = \sum_{j=0}^{\infty} a_j(i) b^{-j-1} \Leftrightarrow i = \sum_{j=0}^{\infty} a_j(i) b^j \qquad (1)$$

where $a_j(i)$ are subsequent digits of i. For instance $\Phi_{10}(1234) = 0.4321$.

For photon shooting we need to associate the index of each photon with a sequence of numbers, thus the multi-dimensional Halton sequence must be considered. Such a sequence is usually obtained by using prime numbers as bases for different dimensions. For example, for integer i the following sequence can be considered:

$$i \Rightarrow \Phi_2(i), \Phi_3(i), \Phi_5(i), \Phi_7(i), \Phi_{11}(i), \ldots \qquad (2)$$

When sampling discrete events, a single quasi-random number can be re-used via the remapping procedure (refer to Shirley et al.[34], Fig. 8). In this way we use the two first quasi-random numbers $\Phi_2(i)$ and $\Phi_3(i)$ to choose a light source and the photon emission direction, the next two numbers $\Phi_5(i)$ and $\Phi_7(i)$ to determine the photon-material event (photon absorption or scattering) and the reflection direction for the first photon bounce, and so on.

4.2. Periodicity of the Halton Sequence

The observation that the Halton sequence has a highly periodic nature is hardly new[29]. Note for example how similar the values of following base-10 radical inverses are: $\Phi_{10}(1234) = 0.4321$, $\Phi_{10}(1234+1000) = 0.4322$, and $\Phi_{10}(1234+2000) = 0.4323$. We express this periodicity property in a form which is well suited for the task of finding similar photons (paths) in photon (path) tracing algorithms:

$$|\Phi_b(i) - \Phi_b(i+mN_g)| < \frac{1}{b^k} : N_g = lb^k \qquad (3)$$

where k, l, and m are integers such that $k \geq 0$ and $l > 0$. Indeed, adding $(ml)b^k$ can only change higher order digits in the base-b representation of i. Namely, $a_j(i)$ with $j \geq k$ can change. Thus, taking into account that the change of a single digit cannot be larger than $(b-1)$ and that only a finite number of digits change, we write:

$$|\Phi_b(i) - \Phi_b(i+(ml)b^k)| < \sum_{j=k}^{\infty} (b-1) b^{-j-1} \qquad (4)$$

Computing the right side of (4) as a sum of infinite geometrical progression, we immediately obtain (3). We further generalize inequality (3) to the case of a multi-dimensional Halton sequence. In the full correspondence with the definition of N_g above, let us set:

$$N_g = b_1^{k_1} b_2^{k_2} \ldots b_n^{k_n} \qquad (5)$$

Substituting such N_g into (3), we derive that N_g yields a good matching of values simultaneously in dimensions $b_1, \ldots, b_n$. According to (3), the matching of value in each particular dimension b is determined by the corresponding k_b. Note also that choosing the sample's number to be a multiple of such defined N_g also results in the discrepancy minimization of the multi-dimensional Halton sequence[9].

5. Algorithm

Our global illumination algorithm operates with a fixed pool of photons, whose size is decided by the user as a trade-off between the desired accuracy of the lighting simulation and the acceptable delay of the illumination update.

In the following section we describe the organization of the photon pool into coherent groups which are processed one by one in our iterative approach. In Section 5.2 the algorithm of illumination update by selective photon group processing is presented. The priority criteria of photon group processing are explained in Section 5.3.

5.1. Photon Groups

All photons in the pool are split into N_g groups, where N_g is selected based on Equation (5). Each group of photons consists of two subgroups. Photon indices in the first subgroup, which we call *the pilot photons*, increase incrementally. Photon indices in the second subgroup, which we call *the corrective photons*, have the interval N_g between them. Figure 1 illustrates how photons are distributed between groups for a case corresponding to $N_g = 4$.

The basic property of all photons from one group is that they are all shot for the same (frozen) scene configuration. For each group we store the following information:

- the group priority which guides the ordering of global illumination update
- a state of dynamic scene properties, valid at the moment the photon group was shot, but not valid at the current

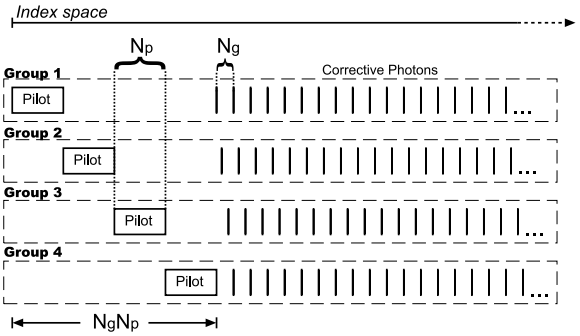


Figure 1: *Distribution of photons between groups for $N_g = 4$. Each row represents one group. Each group contains a number of pilot photons (rectangle) as well as a number of corrective photons (vertical lines) whose indices are spaced with interval N_g.*

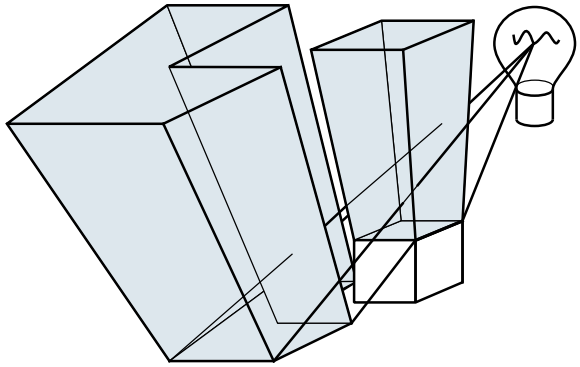


Figure 2: *The bounding volume of corrective photons from a single group starts as a pyramid, but then its shape might become more complicated as the result of photon interaction with scene objects. For the figure clarity we marked just one bounce of photons (grey pyramids reflected from white objects) but obviously all higher bounces are also considered as required for the global illumination computation.*

time moment. We denote such scene properties as *scene phantoms.* The algorithm can consider different dynamic scene properties. The most obvious are object positions and surface BSDFs (Bidirectional Scattering Distribution Functions).

According to the periodicity property discussed in Section 4.2, corrective photons are concentrated in a tight subspace of the multi-dimensional photon space. In practice this means that corrective photons of the same group are usually emitted by the same light source, traced in similar directions, have similar probability of being absorbed when they hit a given surface, and so on. Increasing N_g increases the number of groups and simultaneously decreases the space span occupied by every group. Note that when changing N_g it is important to keep it factorized according to Equation (5). The size and space filling characteristics of photon groups can be tuned according to the desired properties of the global illumination update using the following parameters:

- N_g defines the photon groups number and simultaneously the span between the indices of corrective photons
- N_p defines the number of the pilot photons in each group
- R defines the ratio between the total number of photons in each group and N_p.

Given those parameters, the total number of photons utilized by the algorithm is equal to N_gN_pR. All photons with indices $i \in [0, N_gN_p)$ are the pilot photons, and all photons with indices $i \in [N_gN_p, N_gN_pR)$ are the corrective photons.

5.2. Illumination Update

Our algorithm is fully interactive, which means that it does not require any a priori knowledge on changes in the scene. In the following discussion we assume that user interaction consists of changing object positions (the case of surface BSDF changes is easier to handle and will be explained later). Updating illumination after scene change requires that photons intersecting moving objects must be reshot. Reshooting of a photon consists of tracing it for the old scene configuration with negative energy and then tracing it for the new scene configuration with positive energy. Since usually only a small number of photons intersect moving objects, such double effort is justified compared to calculating the solution anew.

Efficient searching of invalid photons is a hard problem. We attempt to solve it by subdividing the multi-dimensional photon space into relatively small volumes and searching for those volumes, which intersect the moving object. According to the periodicity property discussed in Section 4.2, the corrective photons in a single group which are emitted by a light source are bounded by a pyramid with its apex located at this light source position (Figure 2). The pyramid shape might become more and more complicated as the result of interaction between scene objects and the corrective photons. However, even if the pyramid is split into a number of subpyramids, the resulting bundles of photon paths still remain coherent in space[†] and are fully bounded by those subpyramids.

The computation of intersection between bounding volumes of complex shape (as depicted in Figure 2) and moving

[†] This requires a simplifying assumption that each scene object has roughly uniform light scattering properties across its surface. We believe that this assumption is justified for interactive global illumination solutions which are the main focus of this paper. Obviously this assumption does not hold in general, e.g., for surfaces with a fine pattern of displacement or bump mapping, for surfaces with shift-variant BSDF, and other similar cases.

objects is required to identify invalid photons. Additionally it is important to estimate the impact of those invalid photons on the correctness of rendered images. Such an estimate is required to find out the order in which to reshoot the photon groups, which reduces image artifacts as perceived by the human observer. We address those problems by shooting the pilot photons.

In contrast to the corrective photons, the pilot photons are not coherent and uniformly sample the space of vectors defined in Equation (2). The shooting of pilot photons is interleaved with the shooting of corrective photons by placing them in the same group (see Figure 1). Let i be the pilot photon, which intersected the moving object. We then claim that the bounding volume of corrective photons belonging to group i_g with

$$i_g = i \bmod N_g \tag{6}$$

intersects this object as well. Indeed, all corrective photons of that group are given by indices $N_g N_p + i_g + cN_g$, where c is a non-negative integer. The index i given by Equation (6) differs from the index of any corrective photon of group i_g by some multiple of N_g. Thus, according to Equation (3), this photon is inside of the multi-dimensional pyramid, which bounds all corrective photons of that group. This means that it also is inside of the bounding volume, created when this pyramid intersects scene objects. Since the pilot photons sample the multi-dimensional vector space (defined in Equation (2)) uniformly, the number of their hits with moving objects provides an approximate measure how important it is to update the group (in Section 5.3 we discuss in detail the corresponding problem of photon updates ordering).

Our approach to identify moving objects is not conservative (refer to Briere and Poulin[3] and Bala et al.[1] for discussion of the conservative solutions). Even if no pilot photon detects the intersection of a group of corrective photons with a moving object, it is nevertheless possible that such an intersection takes place. However, this typically happens if the intersected volume is very small and does not contribute much to the changes of global illumination. Also, when scene objects stop their motion, all N_g groups are finally updated, which means that all invalid photons will be corrected, and then the only negative consequence of such undetected intersections is a non-optimal ordering of the group update.

One may advocate that it is easy to conservatively find the intersection between moving objects and the first segment of a pyramid with the apex at the light source position (e.g., such an approach is used to reinforce the caustic photons in the photon mapping technique[18]), however, we are aiming for a general solution that has some potential to detect invalid photons bouncing an arbitrary number of times in the scene. Also, the density and coherence of the pilot photons in the first pyramid segment is very good, which usually results in the high probability of detecting moving objects.

The pseudo-code in Algorithm 1 summarizes the process of illumination update.

Algorithm 1 Global Illumination Update Algorithm.

```
while there are groups containing scene phantoms do
  among all such photon groups, find the one with highest
  priority;
  place scene phantoms of that group into BSP grid and
  forbid scene changes;
  for all photons i in the photon group do
    perform Monte Carlo tracing of photon i according
    to Equation (2) without updating the global illumi-
    nation;
    if photon i hit dynamic object or scene phantom then
      if photon i is pilot then
        if photon i hit dynamic object then
          increase the priority of corresponding group;
        else if photon i hit scene phantom then
          increase the priority of corresponding group
          only if it contains intersected phantom;
        end if
      end if
      duplicate the photon and trace two resulting pho-
      tons:
      - photon, which ignores scene phantoms and has
      positive energy;
      - photon phantom, which ignores dynamic objects
      and has negative energy;
    end if
  end for
  display partially updated solution via OpenGL;
  remove all scene phantoms from the BSP grid and from
  just reshot photon group, allow scene changes;
  set the group priority to 0;
end while
```

Since the concept of scene phantoms is essential for understanding Algorithm 1, we provide an example to explain it in an intuitive way. For simplicity let us assume that a scene contains only one dynamic object and its motion is specified by matrices M_1, M_2, and M_3, which describe three different positions of the object at the moments of time t_1, t_2, and t_3. At t_2 the object position changes from M_1 to M_2, and a new scene phantom is created with the old position M_1. This scene phantom is assigned to all photon groups, and the illumination update, which is required due to the object motion, is performed as specified in Algorithm 1.

For reshooting photons from a given group at t_2, the scene phantom is placed into the BSP grid, which is used to accelerate the photon tracing computation. This means that the BSP grid contains two distinct instances of the moving object at its respective positions M_1 and M_2. Then three types of rays as required by Algorithm 1 are traced: rays which can hit both instances of the moving object, rays which ig-

nore the phantom object at position M_1, and rays that ignore the actual object at position M_2.

At the moment of time t_3 when the object changes its position to M_3, the last update of some photon groups could have been performed at t_2, but there is also a chance that some groups have been updated only at t_1. The latter groups still contain the object phantom at position M_1. Since any photon group corresponds to a frozen scene configuration at a particular moment in the past, it cannot contain two phantoms of the same object. This is why the newly created phantom at position M_2 must be placed only in those photon groups, which do not contain the phantom at position M_1.

User interaction with surface BSDFs is processed in a similar way as described in Algorithm 1. However, this does not involve any update of the ray tracing acceleration data structures. The scene phantoms store the surface BSDFs from the last iteration in which photons from the current group were updated.

In our current implementation user interaction with light sources is not supported, although it is relatively easy to extend our algorithm for this purpose. A number of special cases for changing the light source position, emitted energy, goniometric diagram and so on should be considered. For example, user interaction changing the position of light sources should result in increasing the update priority for all groups for which photons are emitted by dislocated light sources. The old position of the light source should be stored as the scene phantom.

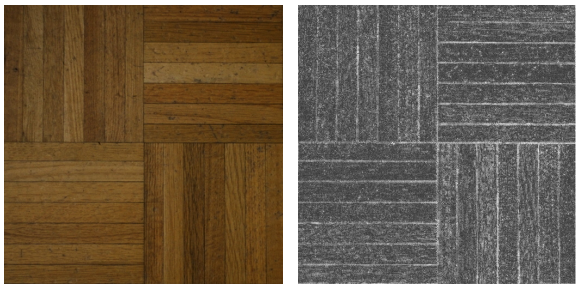
5.3. Priority of Photon Groups Processing

An important issue is setting priorities of photon group processing in order to minimize the perceivability of illumination artifacts by the user. For this purpose we consider three factors which affect the image appearance:

- Changes in the scene detected by the pilot photons.
- Visual masking of lighting patterns by textures.
- Current camera view frustum.

The pilot photons are used to compute the importance of photon groups. If pilot photon i hits a dynamic object, the importance P of the group i_g is increased by increment Δp. Also, if a scene phantom is hit by i and belongs to the list of phantoms for i_g, P is increased by the same increment Δp (some arbitrary value is chosen, e.g, $\Delta p = 1$). The importance increment Δp can be reduced for a given photon hitting a texture as the result of visual masking.

Visual masking of lighting patterns by textures may significantly reduce the observer sensitivity to artifacts in the scene illumination. For surfaces with strong masking, the priority of illumination update can be reduced with respect to non-textured surfaces. We follow the approach by Dumont et al.[10] in which the VDP proposed by Ramasubramanian et al.[32] is used to estimate the visibility threshold elevation T_e for textures at the preprocessing stage. We



a) b)

Figure 3: *Visual masking processing a) sample texture and b) visibility threshold elevation map (brighter regions in the map correspond to higher masking).*

introduce one modification to this procedure which is inspired by Luebke et al.[23]. For a given mesh density and texture mapping parameters for this mesh it is possible to estimate the relation between the highest spatial frequency of indirect illumination reconstructed using this mesh and the spatial frequencies in the texture. The visual masking can be observed mostly between stimuli of similar spatial frequencies[6]. Therefore, prior to the VDP processing we eliminate from the texture all frequencies higher than the mesh imposed maximum frequency of indirect illumination. This modification makes our approach more conservative and excessive values of T_e can be avoided. As a result of the VDP processing we obtain a set of the average T_e values for a given texture at each MipMap level. In Figure 3 (see Appendix/Color section) we show an example of the visibility threshold elevation map obtained using our VDP.

During photon tracing we access textures assigned to scene surfaces in order to estimate the probability of photon absorption and to determine the color of scattered photons. While the texture is sampled at its original resolution, we need to access T_e at the proper MipMap level. For this purpose we use a simple heuristic which takes into account the surface distance to the observer and its spatial orientation. We use the T_e value for the photon hit texture to modify the importance P of the group i_g by increment $\Delta p/T_e$. This is an application example in which the perception-based error metric is used to steer the global illumination computation with negligible overhead costs. Also, the potential inaccuracies of such perceptual steering are fully recoverable in our algorithm because only the computation ordering is affected.

Although our algorithm is view-independent, better efficiency of illumination update can be achieved by assigning higher priorities to the visible regions of the scene. We efficiently combine visibility information with the group importance P by controlling the visibility status V for every photon group. This visibility status is updated for every frame as a by-product of the pilot photons tracing. It is costly to deter-

mine the visibility of the photon hit exactly, thus we only use view frustum culling here. Photon hits inside of view frustum can affect the image quality, so we increase V of the corresponding group. In contrast to the value P, V is view dependent and we have to reset it after each camera change. Thus, V is taken into account only if the number of pilot photons, shot after the last camera change, is sufficient for its robust estimate. Otherwise, we assume that $V = 1$ for each group.

After the pilot photons of the current group are shot, and V and P are updated, the next group of corrective photons with the highest priority measured as $V \cdot P$ is selected.

6. Implementation

We implemented our algorithm as a hybrid rendering system where graphics hardware is responsible for computing direct illumination, the corresponding hard shadows, and visualizing the indirect illumination. The software part of our engine is mostly dedicated to calculating the indirect illumination. Both parts run in parallel using multi-threading, and are synchronized by the important events (update of indirect illumination, user interaction etc.).

For scenes with multiple light sources, whose processing would require many rendering passes of graphics hardware, another strategy of direct lighting computation can be considered. The user may select the most important for his application light sources to be processed by graphics hardware to obtain the good quality of shadows. The remaining light sources can be processed on the photon basis without any extra cost since photons are traced from the light sources anyway. The problem with such an approach is that the direct illumination is then reconstructed on the mesh basis and the resulting shadow boundaries are fuzzy with some possible artifacts.

Shooting photons requires efficient ray tracing acceleration data structures for dynamic environments. Initially we relied on multi-level voxel grids. Recently, we switched to the axis-aligned BSP-tree with a cost function[14], which allowed us to improve the computation performance significantly (up to 3 times). At present we do not use any hardware-dependent optimizations of ray tracing (e.g., SIMD, prefetching, and so on) as suggested by Wald et al.[39]. Obviously by adding such optimizations the ray tracing performance could be further improved.

6.1. Direct Illumination

For computing the effect of direct illumination we chose to use OpenGL-compliant graphics hardware. Instead of restricting ourselves to the fixed function pipeline of standard OpenGL we utilize programmable vertex and pixel hardware to gain the highest qualitative and most efficient rendering possible. Our current implementation is customized to run on NVIDIA GeForce3 graphics cards, but can of course also be implemented on any other card with similar features (e.g. ATI's Radeon series).

6.1.1. Shadows

One of the main aspects of image quality is an accurate representation of shadows. Although shadow mapping[42] is directly supported by the graphics hardware it is problematic due to its sampling problems. Another disadvantage is that for dynamic environments shadow maps change very often which would lead to a huge amount of regeneration passes. We therefore prefer the shadow volume algorithm proposed by Crow[5] since it

- generates very precise shadows by performing calculations in object space,
- can be efficiently implemented using graphics hardware, and
- is appropriate for dynamic environments.

For complex scenes the generation of shadow volumes, which requires finding the silhouette edges of all objects with respect to the light source, is quite expensive. By exploiting temporal coherency of shadow volumes we can limit the regeneration of shadow volumes to those objects that are moving and reuse the volumes of all static objects from the previous frame.

Our shadow volume implementation is based on the hardware stencil buffer scheme presented[21], which solves the problematic cases of shadow volumes intersecting the near clipping plane.

With normal OpenGL the shadow volume algorithm would require $N+1$ rendering passes where N is the number of shadow casting light sources. However using programmable features available on recent graphics hardware[20] we are able to collapse up to 4 passes into a single one. This is done by first rendering the scene's geometry as seen by the camera, resulting in having the depth values of the front most pixels in the depth buffer. After this we loop over the first four light sources where in each step we first initialize the stencil buffer and draw shadow volumes with the corresponding stencil operation (increment/decrement). The content of the stencil buffer, which corresponds to the shadow test result, is then copied to one of the color buffer channels (red channel for first light, green channel for second, etc.) and the stencil buffer is initialized for the next light source. After this loop we obtain a RGBA *shadow mask* containing the shadow information for up to four light sources.

In the final rendering pass we then render the scene once again but this time using a customized vertex program[22] which instead of summing up all lighting contributions for a given vertex uses additional output attributes that hands out the illumination for light source $L_{0\ldots3}$ separately. These values will then be linearly interpolated over the primitive (triangle) and passed to the texture blending stage.

Having the shadow mask as an projective RGBA texture we can apply the shadow result separately for each of the four light sources and output the total illumination as the pixel value.

$$out = indirect + illum(L_0) \cdot mask_R + illum(L_1) \cdot mask_G + \ldots \quad (7)$$

For more light sources this approach can be extended by simple multi-pass rendering. So for N light sources we have to generate $\lceil N/4 \rceil$ shadow masks. Another $\lceil N/4 \rceil$ passes are needed to compute the illumination. These passes can then be summed up using additive blending or the accumulation buffer.

This shadow mask scheme is not only very efficient but also enhances image quality. Using the normal loop scheme, the contributions of all light sources need to be summed up using either the accumulation buffer or additive blending. Since both operations are performed at the end of the pixel pipeline precision is limited to (normally) 8 bits per color channel, whereas our approach sums up at an earlier stage in the pipeline where precision is much higher. Accuracy is also increased for $N > 4$ since we use the accumulation buffer (or additive blending) for groups of four light sources rather than summing up the contribution of individual lights. Another advantage is that our method does not suffer from z-fighting artifacts which normally occur when rendering the scene several times.

6.1.2. Goniometric Diagrams

In order to achieve realistic image quality we also choose to support complex point light sources with non-uniform directional power distribution. Description of these goniometric diagrams are available in standardized formats and are essential for accurate lighting computations.

We include these distributions by re-sampling from the standard format to a cube map texture[38], which can efficiently store the complete 360^o view of a point light source and which is supported by the graphics hardware.

Restricting ourselves to monochromatic goniometric diagrams we can include these as an additional scaling factor for the local illumination of a given light source. Referring to Equation 7 the *mask* scaling factor, which was considered to be either 1.0 for lit and 0.0 for shadowed pixels, can be used to perform this additional scaling. In the first rendering pass we render using the appropriate cube map textures and store the scaling factors in one of the color channels. Using four texture units and RGBA masking we are able to generate these for 4 light sources simultaneously. When copying the result of the shadow test back to the color buffer we set all pixels to 0.0 where the shadow test succeeded. Although the dynamic range of these textures is limited to 0.0 to 1.0 this method still improves image quality significantly while introducing only a minimal overhead.

7. Results

We present results that we obtained for the ROOM scene (about 12,400 mesh elements) and for the HOUSE scene (about 377,900 mesh elements). The ROOM scene is illuminated by four, the HOUSE scene by two light sources with goniometric diagrams. Figures 4 (see Appendix/Color section) show snapshots of the two scenes obtained during an interactive session.

All timings were measured on 1.7 GHz Dual P4 Xeon computer with an NVIDIA GeForce3 64 MB videocard. The frame rate is governed by the speed of OpenGL rendering. For the ROOM scene 8 fps and for the HOUSE scene 1.1 fps are achieved, which includes the display of indirect lighting, and direct lighting with shadow computations performed by the graphics hardware. Our current implementation does not support triangle strip generation which affects the refresh rate. This however is not a limiting factor for the performance of indirect lighting update because it is refined at an even slower pace.

As we mentioned in Section 5.1, our algorithm performance can be tuned by the parameters N_g, N_p and R. The parameter N_g controls the number of the Halton sequence dimensions, for which corrective photons of the same group remain coherent. We chose the initial factorization of N_g for two dimensions to be $2^3 3^2$. This splits the first and second dimensions to 8 and 9 spans, respectively. The remapping procedure (refer to Section 4.1), applied after the light source selection step, effectively reduces the spatial coherence in the first dimension. To compensate for this, we increase the corresponding power k_2 proportionally to the number of light sources present in the scene. This results in factorizations $2^5 3^2$ and $2^4 3^2$ for the ROOM and HOUSE scenes, respectively.

The graphs in Figure 5 demonstrate the influence of N_g parameter on the speed of the global illumination update as the result of an object movement from one position to another. The efficiency of the update is measured by the number of reshot *relevant photons* as a function of time. We call those photons relevant, whose hit points are dislocated as a result of scene changes. All relevant photons must be reshot to complete the illumination update. The algorithm performance is compared for $N_g = 288$ (factorization: $2^5 3^2$) and $N_g = 10080$ (factorization: $2^5 3^2 5^1 7^1$). For $N_g = 288$ only direct photons belonging to a group are spatially coherent, while for $N_g = 10080$ also the first bounce of scattered photons is coherent. Two cases are considered for each of the N_g values: the moving object is illuminated 1) directly and 2) indirectly. In the first case the number of groups has a small impact on the performance of illumination update. This can be explained by the fact that the moving object is mostly hit by the direct photons, which are coherent even for small N_g. If the moving object is illuminated indirectly, a much better performance is obtained for larger N_g because the coherence

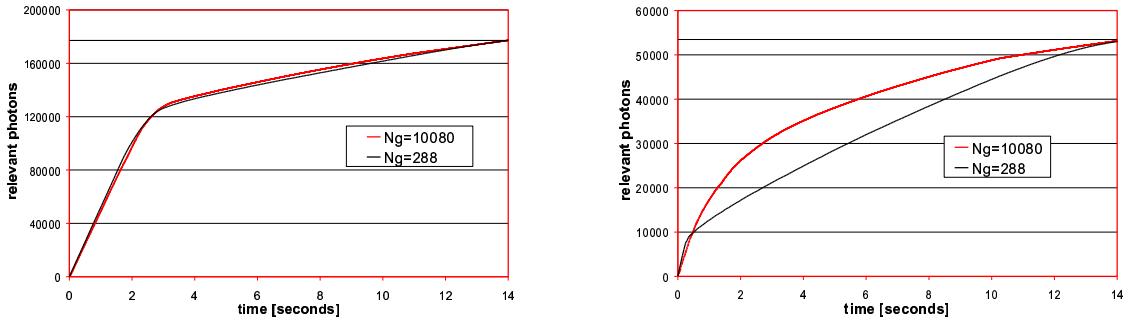


Figure 5: *The number of found relevant photons as a function of time. The moving object is illuminated by a) both direct and indirect lighting, b) only indirect lighting. The horizontal line shows the total number of relevant photons, which must be updated.*

of the first bounce of reflected photons becomes important. Such coherence is not ensured for $N_g = 288$.

Figure 6 (see Appendix/Color section) shows the ability of the algorithm to find the coherent photons after the first indirect bounce from the surface. Photons from both depicted groups reflect diffusely from the floor in the direction of the moving objects.

The speed of the global illumination update depends strongly on the pilot photon's ratio in the pool, which defines the precision of the group priority computation. If this ratio is too small, important photon groups can be missed and then reshot too late. Conversely, if the number of pilot photons is too large, the update efficiency is reduced because the pilot photons are shot essentially in a random order. Figure 7 demonstrates this dependency for the ROOM scene. N_p and R values are selected in such a way, that the total number of photons is constant and only the ratio of pilot photons changes. We have found that $R = 16$, which corresponds to only 6.25% of pilot photons in the pool, yields nearly optimal performance. As can be clearly seen in Figure 7, for $R = 32$ (3.125% of pilot photons) the photon group ordering is not robust and the overall update performance is worse.

According to our experiments, full recovery of perceivable illumination artifacts resulting from the furniture movements in the ROOM and HOUSE scenes usually takes 2–4 and 4–8 seconds, respectively. The total number of photons in the pool for the ROOM scene was 1,152,000 and for the HOUSE scene 1,728,000.

Storing the photon paths for all photons could be manageable for the number of photons we presently use, which could perhaps help to avoid some computation, e.g., tracing some photons with negative energy. However, our experiments have shown that tracing changed photon paths twice introduces an overhead which on average was below 25% (some initial segments of the photon path may be the same

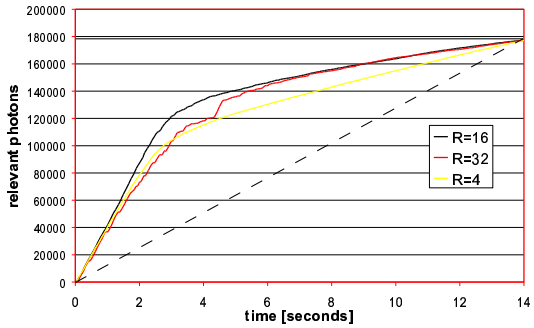


Figure 7: *Number of found relevant photons as a function of time, depending on the ratio R between the total number of photons and the number of pilot photons. The total number of photons in the pool is constant. The dotted line shows the theoretical update progress under the assumption that the photons are updated in a random order.*

and can be computed only once). Handling stored photon paths would incur some overhead, and the extension of our algorithm to handle more photons and a finer mesh in order to obtain an even better quality of lighting reconstruction would be more difficult (e.g. more than one processor for photon tracing could be used or a faster ray tracing algorithm[39] could be implemented).

8. Conclusions

We propose a novel interactive global illumination technique for dynamic environments which is suitable for processing complex scenes. Based on the periodicity property of the Halton sequence we are able to trace photons coherently only into those scene regions which require the illumination

update. Using our technique, the temporal coherence of the scene illumination can efficiently be exploited without any significant storage of data involving the temporal domain. Progressive refinement of rendered frames is steered using energy- and perception-based criteria in order to minimize image artifacts as perceived by the user. As a result subsequent frames can be efficiently rendered and the temporal aliasing is practically reduced below the perceivability level. The quality of the direct lighting distribution and shadows significantly benefit from our efficient hardware implementation.

Selective photon tracing has many potential applications which require local reinforcement of computations based on some importance criteria. An example of such an application is the efficient rendering of high quality caustics, which usually requires a huge number of photons. After identifying some paths of caustic photons, more coherent caustic photons can easily be generated using our approach. The drawback of many existing photon based methods is that too many photons are sent into well illuminated scene regions with a simple illumination distribution, and too few photons reach remote scene regions. This deficiency could easily be corrected using our approach, by skipping the tracing of redundant photons and properly scaling the energy of the actually traced photons. As future work we plan to pursue our research along those avenues. Also, we would like to include specular and glossy effects into our interactive renderer using pixel-selective ray tracing.

9. Acknowledgments

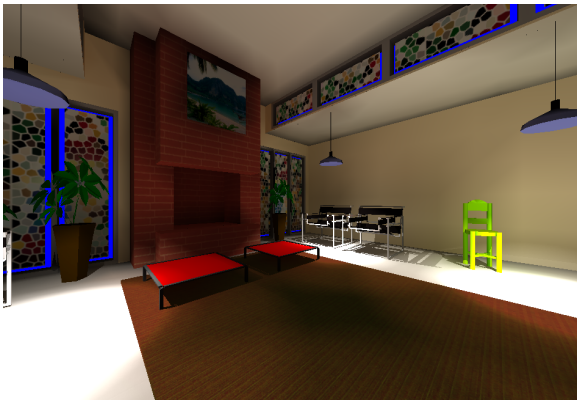
We would like to thank Philippe Bekaert, Katja Daubert and Vlastiml Havran for their helpful comments and suggestions. This work was supported in part by the European Community within the scope of the RealReflect project IST-2001-34744 “Realtime visualization of complex reflectance behavior in virtual prototyping”.

References

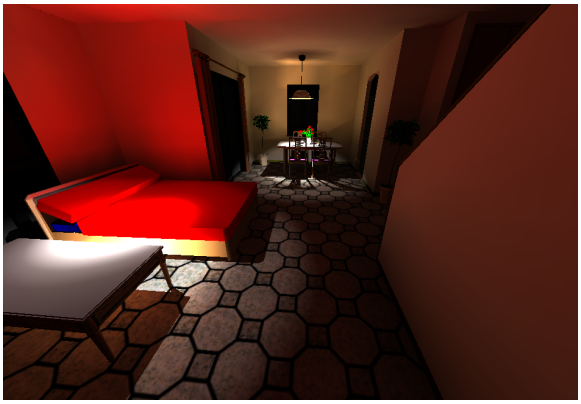
1. K. Bala, J. Dorsey, and S. Teller. Radiance interpolants for accelerated bounded-error ray tracing. *ACM Transactions on Graphics*, 18(3):213–256, 1999. 6
2. G. Besuievsky and X. Pueyo. Animating radiosity environments through the multi-frame lighting method. *The Journal of Visualization and Computer Animation*, 12(2):93–106, 2001. 2
3. H. Briére and P. Poulin. Hierarchical view-dependent structures for interactive scene manipulation. In *Proceedings of SIGGRAPH 96*, Computer Graphics Proceedings, Annual Conference Series, pages 83–90, 1996. 6
4. S.E. Chen. Incremental Radiosity: An Extension of Progressive Radiosity to an Interactive Image Synthesis System. In *Computer Graphics (SIGGRAPH 90 Conference Proceedings)*, pages 135–144, 1990. 2
5. F.C. Crow. Shadow algorithms for computer graphics. In *Computer Graphics (SIGGRAPH ’77 Proceedings)*, pages 242–248, July 1977. 8
6. S. Daly. The Visible Differences Predictor: An algorithm for the assessment of image fidelity. In A.B. Watson, editor, *Digital Image and Human Vision*, pages 179–206. Cambridge, MA: MIT Press, 1993. 7
7. C. Damez and F.X. Sillion. Space-Time Hierarchical Radiosity for High-Quality Animations. In *Eurographics Rendering Workshop 1999*, pages 235–246, 1999. 2
8. G. Drettakis and F.X. Sillion. Interactive Update of Global Illumination Using a Line-Space Hierarchy. In *SIGGRAPH 97 Conference Proceedings*, Annual Conference Series, pages 57–64, 1997. 2, 3
9. Th. Driemeyer and R. Herken. *Programming mental ray*. Springer, page 226, 2000. 4
10. R. Dumont, F. Pellacini, and J. Ferwerda. A perceptually-based texture caching algorithm for hardware-based rendering. In *Eurographics Workshop on Rendering 2001*, pages 249–256, 2001. 7
11. D.W. George, F.X. Sillion, and D.P. Greenberg. Radiosity Redistribution for Dynamic Environments. *IEEE Comp. Graphics and Applications*, 10(4):26–34, 1990. 2
12. X. Granier and G. Drettakis. Incremental updates for rapid glossy global illumination. *Computer Graphics Forum*, 20(3):268–277, 2001. 2
13. John H. Halton. *On the efficiency of certain quasi-random sequences of points in evaluating multi-dimensional integrals*. Numerische Mathematik, 2:84-90, 1960. 4
14. V. Havran. *Heuristic Ray Shooting Algorithms*. PhD thesis, Czech Technical University, Nov. 2000. 8
15. P. Heckbert. Adaptive Radiosity Textures for Bidirectional Ray Tracing. In *Computer Graphics (SIGGRAPH 90 Conference Proceedings)*, pages 145–154, August 1990. 2, 3
16. S. Heinrich and Alexander Keller. Quasi-Monte Carlo Methods in Computer Graphics Part II: The Radiance Equation. Technical Report 243/94, 1994. 3
17. E. Hlawka. *Discrepancy and Riemann Integration, Studies in Pure Mathematics (New York)*. 1971. 3
18. H.W. Jensen. *Realistic Image Synthesis Using Photon Mapping*. AK, Peters, 2001. 2, 6
19. A. Keller. Instant Radiosity. In *SIGGRAPH 97 Con-*

ference Proceedings, Annual Conference Series, pages 49–56, 1997. 2, 3, 4

20. M.J. Kilgard. *NVIDIA OpenGL Extension Specifications*. NVIDIA Corporation, November 2001. Available from http://www.nvidia.com. 8
21. M.J. Kilgard. Robust stencil shadow volumes, 2001. Available from http://www.nvidia.com. 8
22. E. Lindholm, M.J. Kilgard, and H. Moreton. A user-programmable vertex engine. *Proceedings of SIGGRAPH 2001*, pages 149–158, August 2001. 8
23. D. Luebke and B. Hallen. Perceptually-driven simplification for interactive rendering. In *Eurographics Workshop on Rendering 2001*, pages 223–234, 2001. 7
24. I. Martín, X. Pueyo, and D. Tost. Frame-to-frame coherent animation with two-pass radiosity. Technical Report (To appear in *IEEE Transactions on Visualization and Computer Graphics)* 99-08-RR, GGG/IIIiA-UdG, Jun 1999. 2
25. S. Müller, W. Kresse, and F. Schoeffel. A Radiosity Approach for the Simulation of Daylight. In *Eurographics Rendering Workshop 1995*, pages 137–146, 1995. 2
26. K. Myszkowski. Lighting reconstruction using fast and adaptive density estimation techniques. In *Eurographics Rendering Workshop 1997*, pages 251–262, 1997. 2
27. K. Myszkowski, T. Tawara, H. Akamine, and H-P. Seidel. Perception-guided global illumination solution for animation rendering. In *SIGGRAPH 2001*, Annual Conference Series, pages 221–230, 2001. 2
28. L. Neumann, A. Neumann, and P. Bekaert. Radiosity with well distributed ray sets. *Computer Graphics Forum*, 16(3):261–270, August 1997. 4
29. H. Niederreiter. *Random Number Generation and Quasi-Monte Carlo Methods.* Chapter 4, SIAM, Pennsylvania, 1992. 1, 3, 4
30. J. Nimeroff, J. Dorsey, and H. Rushmeier. Implementation and Analysis of an Image-Based Global Illumination Framework for Animated Environments. *IEEE Transactions on Visualization and Computer Graphics*, 2(4):283–298, 1996. 2
31. X. Pueyo, D. Tost, I. Martin, and B. Garcia. Radiosity for Dynamic Environments. *The Journal of Visualization and Comp. Animation*, 8(4):221–231, 1997. 2, 3
32. M. Ramasubramanian, S.N. Pattanaik, and D.P. Greenberg. A perceptually based physical error metric for realistic image synthesis. In *SIGGRAPH 99*, Annual Conference Series, pages 73–82, 1999. 7
33. F. Schoeffel and P. Pomi. Reducing Memory Requirements for Interactive Radiosity Using Movement Prediction. In *Eurographics Rendering Workshop 1999*, pages 225–234, 1999. 2, 3
34. P. Shirley, C. Wang, and K. Zimmerman. Monte Carlo techniques for direct lighting calculations. *ACM Transactions on Graphics*, 15(1):1–36, January 1996. 4
35. M. Simmons and C.H. Séquin. Tapestry: A dynamic mesh-based display representation for interactive rendering. In *Eurographics Rendering Workshop 2000*, pages 329–340, 2000. 3
36. M. Stamminger, J. Haber, H. Schirmacher, and H-P. Seidel. Walkthroughs with corrective texturing. In *Eurographics Rendering Workshop 2000*, pages 377–390, 2000. 3
37. V. Volevich, K. Myszkowski, A. Khodulev, and E.A. Kopylov. Using the Visible Differences Predictor to Improve Performance of Progressive Global Illumination Computations. *ACM Transactions on Graphics*, 19(2):122–161, 2000. 3
38. D. Voorhies and J. Foran. Reflection vector shading hardware. *Proceedings of SIGGRAPH 94*, pages 163–166, 1994. 9
39. I. Wald, P. Slusallek, and C. Benthin. Interactive distributed ray tracing of highly complex models. In *Rendering Techniques 2001: 12th Eurographics Workshop on Rendering*, pages 277–288, 2001. 8, 10
40. B. Walter, G. Drettakis, and S. Parker. Interactive Rendering using the Render Cache. In *Eurographics Rendering Workshop 1999*, pages 19–30, 1999. 3
41. B.J. Walter, P.M. Hubbard, P. Shirley, and D.P. Greenberg. Global illumination using local linear density estimation. *ACM Transactions on Graphics*, 16(3):217–259, 1997. 2, 3
42. L. Williams. Casting curved shadows on curved surfaces. In *Computer Graphics (SIGGRAPH '78 Proceedings)*, pages 270–274, August 1978. 8

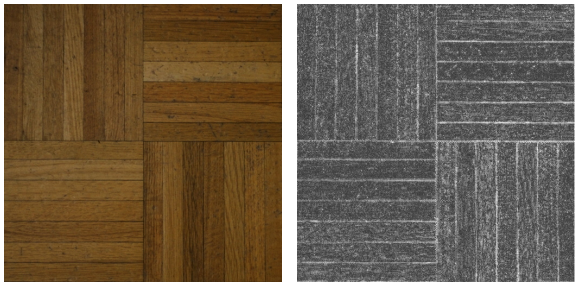


a)



b)

Figure 4: *Interactive session snapshots for scenes a)* ROOM *and b)* HOUSE.



a)

b)

Figure 3: *Visual masking processing a) sample texture and b) visibility threshold elevation map (brighter regions in the map correspond to higher masking).*

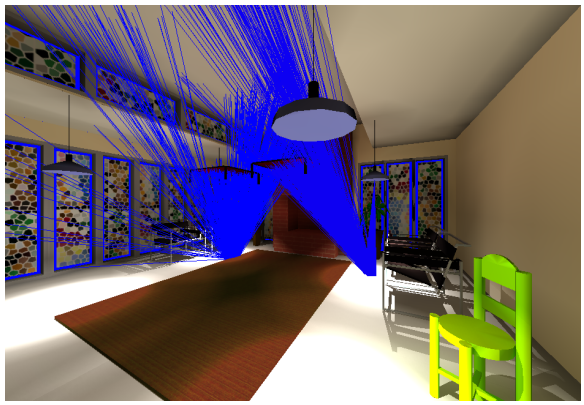


Figure 6: *Two different groups of coherent photons updating the illumination of moving objects (two red tables suspended in the air). Note that for the figure clarity only two photon bounces are depicted while obviously the higher order bounces of photons are traced as required for the global illumination computation.*