Eurographics Symposium on Rendering 2008
Steve Marschner and Michael Wimmer
(Guest Editors)

Volume 27 (2008), Number 4

Exploiting Visibility Correlation in Direct Illumination

Petrik Clarberg and Tomas Akenine-Möller[†]

Lund University

Abstract

The visibility function in direct illumination describes the binary visibility over a light source, e.g., an environment map. Intuitively, the visibility is often strongly correlated between nearby locations in time and space, but exploiting this correlation without introducing noticeable errors is a hard problem. In this paper, we first study the statistical characteristics of the visibility function. Then, we propose a robust and unbiased method for using estimated visibility information to improve the quality of Monte Carlo evaluation of direct illumination. Our method is based on the theory of control variates, and it can be used on top of existing state-of-the-art schemes for importance sampling. The visibility estimation is obtained by sparsely sampling and caching the 4D visibility field in a compact bitwise representation. In addition to Monte Carlo rendering, the stored visibility information can be used in a number of other applications, for example, ambient occlusion and lighting design.

Categories and Subject Descriptors (according to ACM CCS): I.3.7 [Computer Graphics]: Color, shading, shadowing, and texture; I.3.7 [Computer Graphics]: Ray tracing; G.3 [Probability and Statistics]: Probabilistic algorithms (including Monte Carlo)

1. Introduction

In photo-realistic rendering, the lighting is often divided into direct and indirect illumination, which are evaluated separately. The *indirect* illumination is typically low-frequency, as it is the accumulated result of a long (infinite) series of bounces, and efficient algorithms for sparsely sampling and reusing indirect illumination exist. The *direct* illumination is given by an integral over the incident lighting, surface reflectance, and *visibility* [Kaj86]. As we will see in Section 4, the visibility is often strongly correlated between nearby points. Our goal is to exploit this correlation to obtain faster Monte Carlo (MC) rendering with higher quality.

The direct illumination often shows high-frequency features such as sharp shadows, bright specular reflections, and so on. Due to this high-frequency behavior, the algorithms used for indirect illumination, e.g., photon mapping [Jen01], (ir)radiance caching [WRC88, KGPB05, GBP07], and statistical PCA-based filtering [MA06], are not directly applicable. These algorithms essentially perform a low-pass filtering of the radiance field, which may lead to visible artifacts such as blurred shadows when applied to direct illumination.

Current state-of-the-art methods for Monte Carlo evaluation of direct illumination are based on importance sampling the product of lighting and reflectance [BGH05, TCE05, CJAMJ05, CETC06, CAM08]. However, none of these methods take visibility into account. A simple example where product sampling gives poor results is a glossy surface reflecting a bright light source, which is partially occluded. Product sampling directs most samples toward the occluded light, but occasionally a sample hits the light, resulting in a locally high noise level. An example is shown in Figure 1.

We propose a simple, yet efficient, method to exploit correlation in the visibility function to improve the results. Our approach is based on *control variates* [KW86], which is a classic MC technique. The idea is to subtract a correlated approximation from the function we want to integrate, and thus reduce the variance of the remaining part, as illustrated in Figure 2. The difference between the two functions is sampled, and the integral of the approximation is added back as a correction term. In our case, we estimate the visibility from

[†] {petrik|tam}@cs.lth.se

Figure 1: *Techniques for importance sampling only the product of the lighting and BRDF suffer, in general, from excessive noise in occluded regions (left). By adding a control variate term taking visibility into account, we can significantly reduce the problem (right). Both images are unbiased and rendered using 10 samples/pixel.*

nearby samples, and use the triple product of lighting, reflectance and visibility as a control variate term.

Our technique presents an unbiased way of reusing visibility information to improve the rendering quality. The key question is how to obtain the visibility approximation. We have opted for a strategy inspired by radiance caching [KGPB05]. We sample the 4D visibility field sparsely over all visible surfaces, and store the information as 2D visibility maps at selected positions. These maps are interpolated to yield a visibility approximation for the current pixel. It is important to stress that, although we cache visibility data, the primary purpose is to reduce the number of shader evaluations. By spending some extra effort on evaluating the visibility (which is relatively cheap), we reduce the number of samples needed in the importance sampling step, thus keeping the number of expensive shader evaluations at a minimum.

2. Related Work

The method of control variates, or *correlated sampling*, has been used in several computer graphics papers. Lafortune and Willems [LW94] used a constant ambient term as control variate. Later, they stored an approximation of the incident radiance in a 5D tree [LW95] in order to reduce the variance in path tracing. Szirmay-Kalos et al. [SKCA01] used a radiosity solution as a control variate for a subsequent Monte Carlo step. Szécsi et al. [SSSK04] combine correlated and importance sampling to improve the quality of direct illumination. This approach is similar to ours, but their approximation ignores visibility and assumes a diffuse BRDF. Fan et al. [FCH*06] combine samples from multiple functions, which are used as control variates. This can be seen as a generalization of multiple importance sampling [VG95].

A few techniques for exploiting coherence in Monte Carlo evaluation of direct illumination exist. Ghosh et al. [GH06b] exploit *temporal* coherence in the illumination to efficiently render scenes under animated environment map lighting. Samples are generated for the first frame using traditional methods, and then updated for subsequent frames using a sequential Monte Carlo sampling strategy. In other work, Ghosh and Heidrich [GH06a] target *spatial* coherence in visibility. First, bidirectional importance sampling is used to find pixels that are partially occluded. In a second step, they apply Metropolis-Hastings mutations to reduce the noise in these regions. This method only works for occluded regions. In contrast, our approach lowers the variance overall, and exploits both temporal and spatial coherence. Donikian et al. [DWB*06] divide the image into 8×8 blocks, and use adaptive importance sampling to iteratively refine probability density functions (pdf) for the blocks and pixels. In early iterations, more emphasis is put on the block pdf, thereby automatically exploiting spatial coherence. Their algorithm has many clever twists, but since they start without prior knowledge of the lighting and BRDF, many shadow rays are needed (often above 1000 rays/pixel).

Methods for reducing the number of shadow tests have a long history in computer graphics (see, e.g., [War91,SWZ96, FBG02]). The goal of these algorithms is to select a small set of representative lights out of a large number of light sources, usually by means of sorting or spatial data structures, but without taking the BRDF into account. Recent algorithms for many-light rendering [WFA*05, WABG06] share the same goal, but do not explicitly exploit visibility correlation.

Hart et al. [HDG99] use lazy visibility evaluation to compute direct illumination in scenes with many area light sources. They build a "blocker map" for each pixel, exploiting spatial visibility coherence by a flood-fill algorithm in screen space. Similarly, Agrawala et al. [ARHM00] exploit image-space correlation to render soft shadows from area light sources. Ben-Artzi et al. [BARA06] recently presented a clever method for reducing the number of shadow rays needed for environment map illumination. They partition the environment map into a set of preintegrated lights, and use a coarse-to-fine evaluation of the image. By sharing occluder information between neighboring pixels and directions, they can significantly reduce the number of rays traced. These methods are somewhat similar to our visibility cache, but do not exploit temporal coherence. We also use the original unapproximated environment map.

The main difference to previous methods is that we try to solve the combined problem, taking both reflectance, lighting, and visibility into account, while exploiting visibility correlation. Our algorithm builds on existing work on importance sampling and is an *unbiased* Monte Carlo technique, which means it can be used for reference renderings and with very high-resolution environment maps. Our algorithm also has a broader applicability, and can be used for efficient *ambient occlusion* and *lighting design*, i.e., fast preview of complex lighting and shadowing. An important contribution is also our statistical analysis of the visibility function.

3. Mathematical Foundation

The outgoing radiance, L_o, due to direct illumination is given by an integral over the incident lighting, the reflectance (BRDF times a cosine-term), and the visibility [Kaj86]. We denote these terms L, B, and V, respectively. The traditional Monte Carlo estimator for L_o using importance sampling is:

$$\langle L_o \rangle = \frac{1}{N} \sum_{i=1}^{N} \frac{LBV}{p(\omega)}, \tag{1}$$

where $p(\omega)$ is the importance function. Note that we have omitted the arguments of L, B and V for clarity. In the following, we denote approximations of the exact functions by adding a tilde, e.g., $\tilde{B}$. Several recent sampling algorithms explicitly compute and sample an approximation of the product LB. For example, in two-stage importance sampling [CETC06] and quadtree-based product sampling [CAM08], hierarchical approximations of the reflectance, $\tilde{B}$, are multiplied with the exact lighting L, i.e., $p(\omega) \propto L\tilde{B}$.

We propose to include a binary estimation of the visibility, $\tilde{V}$, to lower the variance. A natural approach would be to use the triple product $L\tilde{B}\tilde{V}$ for importance sampling. However, this requires $\tilde{V} \neq 0$ wherever $V \neq 0$, as zeroes in $\tilde{V}$ effectively stop the exploration of those regions of the integral and may lead to bias. Since we do not know which parts of V are truly zero, $\tilde{V}$ has to be nonzero over the entire domain to guarantee correctness. One approach would be to add a small constant, ε, and sample according to $\tilde{V} + \varepsilon$. However, in regions where $\tilde{V} = 0$, but $V = 1$, we would get excessive noise as we divide by $p(\omega)$ in Equation 1, which in this case is very small. We avoid these problems by sampling according to the product $p(\omega) \propto L\tilde{B}$ as before, and use $L\tilde{B}\tilde{V}$ as a *control variate*. Equation 1 is rewritten as follows:

$$\langle L_o \rangle = \frac{1}{N} \sum_{i=1}^{N} \frac{LBV - \alpha L\tilde{B}\tilde{V}}{p(\omega)} + \alpha \underbrace{\int L\tilde{B}\tilde{V} d\omega}_{J}. \tag{2}$$

It is easy to show that this is an unbiased estimator for L_o, as the MC evaluation of $\alpha L\tilde{B}\tilde{V}$ converges to αJ as $N \to \infty$.

With existing importance sampling methods that already compute hierarchical representations of $L\tilde{B}$ [CETC06, CAM08], including a third term, $\tilde{V}$, in the product is relatively inexpensive. More details will be given in Section 5. One interpretation of Equation 2 is that we compute a rough estimate, J, of the direct illumination based on approximations, and then evaluate the *difference* between this and the correct solution using Monte Carlo integration. The rest of this paper deals with the computation of $\tilde{V}$.

3.1. Variance Analysis

In the following, we assume that our algorithm is implemented on top of a scheme for importance sampling the product of lighting and reflectance. In practice, we use the quadtree-based method of Clarberg et al. [CAM08]. For this case $p(\omega) = L\tilde{B}/L_{ns}$, where L_{ns} represents the unoccluded illumination, and Equation 2 can be rewritten:

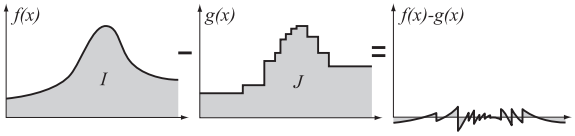


Figure 2: *The idea behind control variates is to subtract a function, g, with known integral, J, from the function, f, we want to integrate, and thus reduce the variance of the remaining part, $f - g$.*

$$\langle L_o \rangle = \frac{L_{ns}}{N} \sum_{i=1}^{N} \underbrace{\left[\frac{B}{\tilde{B}} V - \alpha \tilde{V} \right]}_{Y} + \alpha J. \tag{3}$$

We use the variables Y as defined above, $Z = BV/\tilde{B}$ and $W = \tilde{V}$ to represent random observations of the respective functions, and note that the variance of the estimator in Equation 3 is equal to: $L_{ns}^2 \sigma_Y^2 / N$, where σ_Y^2 denotes the variance of Y given by:

$$\sigma_Y^2 = \sigma_Z^2 + \alpha^2 \sigma_W^2 - 2\alpha \sigma_{ZW}^2. \tag{4}$$

Here, σ_{ZW}^2 denotes the *covariance* of Z and W. The variance is minimized when $\alpha = \sigma_{ZW}^2 / \sigma_W^2$, in which case:

$$\sigma_Y^2 = \sigma_Z^2 (1 - \rho^2), \tag{5}$$

where ρ is the statistical correlation (Pearson's product-moment coefficient) between Z and W. This is defined as follows:

$$\rho = \frac{\sigma_{ZW}^2}{\sigma_Z \sigma_W}, \qquad -1 \leq \rho \leq 1. \tag{6}$$

The correlation coefficient, ρ, is a convenient measure of similarity, as it always lies in the range $[-1, 1]$, where $\rho = 1$ indicates a positive linear relationship, $\rho = -1$ a negative linear relation, and values in-between indicate a weaker correlation.

Equation 5 implies that, assuming we know the value of α, introducing a control variate term reduces the variance of the original estimator [CAM08] by up to a factor $1 - \rho^2$. Using control variates is attractive since the variance goes to zero when the correlation ρ is ± 1, while in the worst case there is no correlation $\rho = 0$, and the variance is unchanged.

However, finding the optimal α is difficult in practice, as the values of σ_{ZW}^2 and σ_W^2 are unknown. The variance of the visibility approximation can be computed, but σ_{ZW}^2 has to be estimated based on samples. In our tests, the additional computation was not motivated by a large enough quality improvement. In addition, bias is introduced if the same samples are used for estimating σ_{ZW}^2 as for evaluating the integral. Instead, we use $\alpha = 1$, which works well because $\tilde{V}$ is reasonably close to $BV/\tilde{B}$.

4. The Visibility Function

The visibility function, V, measures the visibility between a point, $\mathbf{x}$, and a light source, L. This can be written as a function of world-space direction, ω, as follows:

$$V(\mathbf{x},\omega) = \begin{cases} 1, & \text{if } L \text{ is visible,} \\ 0, & \text{otherwise.} \end{cases} \quad (7)$$

By definition, the lower hemisphere is always occluded. If we restrict $\mathbf{x}$ to the surfaces of a scene, V becomes a four-dimensional function. Our goal is to exploit coherence in V to improve the rendering quality. An important question is, how much coherence is there in a typical scene? To answer this, we have performed extensive statistical measurements.

4.1. Correlation Measurements

Intuitively, looking at two random points, $\mathbf{x}_i$ and $\mathbf{x}_j$, the correlation between $V_i = V(\mathbf{x}_i,\omega)$ and $V_j = V(\mathbf{x}_j,\omega)$ should be stronger the smaller the distance, $d = ||\mathbf{x}_i - \mathbf{x}_j||$, is between them. An important factor is also the angular difference, $\theta = \cos^{-1}(n_i \cdot n_j)$, between the surface normals at the two points. As the lower hemisphere of V is always occluded, the similarity between V_i and V_j is likely to be smaller if the surface frames are rotated relative to each other.

We have estimated the *average correlation*, $\bar{\rho}$, as a function of distance and normal difference, i.e., $\bar{\rho}(d,\theta)$, on a suite of test scenes. This was done by distributing a set of sample positions (about 700k) over the visible surfaces, and then measuring the correlation between the visibility functions using a large number of randomly chosen pairs of positions, $\{\mathbf{x}_i,\mathbf{x}_j\}$. Each measurement gives an estimate of ρ for a specific d and θ. These were stored in bins representing small discretized intervals of d and θ. Finally, for each bin, the mean and standard deviation of the correlation estimates were computed. In total, $1.9-3.2\cdot10^{10}$ correlation estimates per scene were done, each based on 96×96 visibility samples over the sphere. The results are summarized in Figure 3 and 4.

4.2. Test Scenes

Four different test scenes were used (see Figure 4). The scenes (a) and (b) represent different camera angles in a garden scene (available on the Autodesk Maya 2008 DVD) tessellated to about 2 million triangles. This scene features extremely difficult occlusion due to the small and highly detailed geometry. The third scene (c) is simpler, although it contains a number of plants not visible in the image, while the last (d) is even simpler with mostly flat surfaces. As we measure the correlation as a function of world space distance, it is important to know the relative sizes of objects. In (a) and (b), the tulips have a diameter of about 0.8 units, while the average height of the grass is 3.8 units. The mushrooms in (c) have a height and diameter of about 1.3, while the cubes in (d) have a side length of 5.0 units.

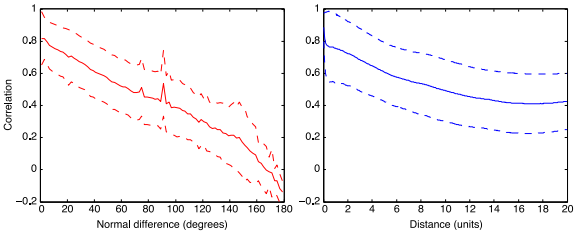


Figure 3: *Analysis of the visibility correlation in scene (a) from Figure 4. All pairs of measurement points were sorted into bins based on their distance and normal difference. The left graph shows the correlation for points that are nearby in distance ($d < 0.2$), and the right graph shows the correlation for points nearby in direction ($\theta < 2^\circ$). The dashed lines represent the standard deviation ($\pm\sigma$).*

4.3. Discussion

The average correlation, $\bar{\rho}(d,\theta)$, provides an interesting footprint of the statistical properties of V. In scene (d), for example, the geometry mainly consists of flat surfaces, i.e., the cubes' faces. Each combination of two such surfaces gives a distinct horizontal line in the correlation plot, as θ is constant, and the range of d is limited by the location and extent of the two surfaces. As expected, the overall correlation is smaller in scenes with "difficult" visibility. However, even for the garden scene, there is a significant amount of correlation between nearby locations. In general, the visibility at surface points with similar normals is highly correlated, even when they are relatively far from each other.

To visualize the spatial distribution of the visibility correlation, we compute the average correlation within a local neighborhood around each point. Only visibility samples within a distance $d_{\max}$, and with $\theta < \theta_{\max}$ were considered. The values of $d_{\max}$ and $\theta_{\max}$ were set as to include mainly the part of $\bar{\rho}$ where the correlation is high (red/orange). This can be seen as cutting out the top-left part of the correlation plot, and computing its average value at different positions in the scene.

Our results show that the visibility is often highly correlated over large smooth surfaces. More interestingly, we found that the correlation can be high even in places of very complex geometry, e.g., the grass in the background in (a) and deep inside the vegetation in (b). In some parts of (c) and (d), the average correlation is surprisingly low, even on smooth surfaces. This happens when the search distance, $d_{\max}$, is too large and samples from nearby surfaces with different visibility are included in the average. This shows that, in order to approximate the visibility function from nearby samples, a uniform distribution is not enough. We must be able to refine the approximation where needed.

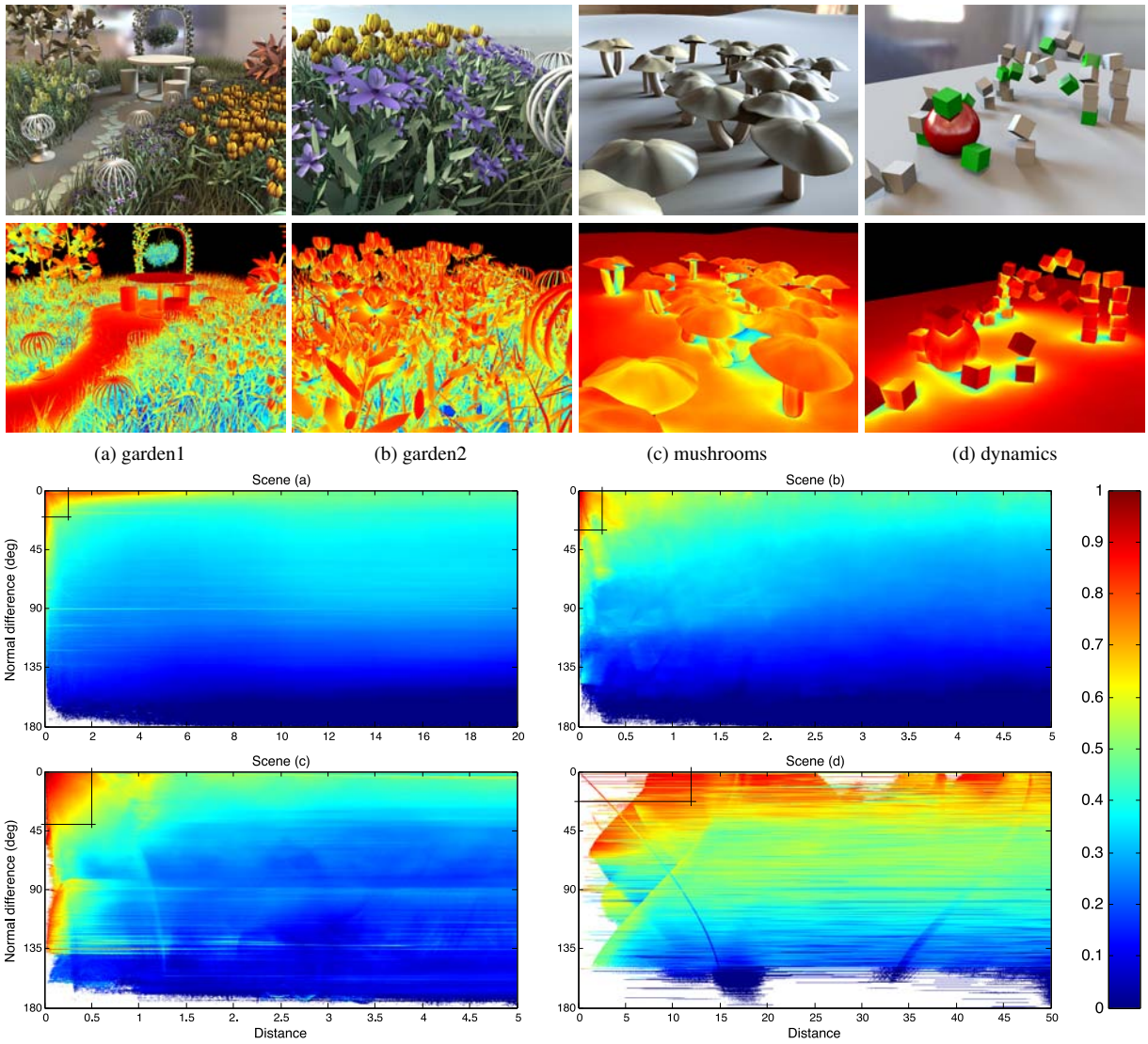


Figure 4: *Measurements of the visibility correlation in four different scenes (a)–(d). All correlation values have been clamped to* $\rho \in [0,1]$ *to make the differences more noticeable, as the correlation is rarely below 0. The bottom four plots show the average correlation as a function of normal difference,* θ*, and world space distance,* d*. White represents combinations* (d,θ) *for which there was not enough data. In total, about* $1.7-3.0\cdot 10^{14}$ *pairs of visibility samples were considered for each image. As expected, the correlation is close to 1 in the top-left corner. The four false-color renderings above show the spatial distribution of the visibility correlation. For each pixel, we have computed the average correlation to all nearby points within a certain distance and normal difference,* $(d_{\max},\theta_{\max})$*, represented by a box in the correlation plots. The reader is encouraged to zoom in and examine the images in detail.*

5. Exploiting Coherence: The Visibility Cache

Our goal is to construct an approximation, $\tilde{V}$, which is as close to V as possible. For this purpose, we evaluate V at a sparse set of locations $\{\mathbf{x}_i\}$, and store the resulting 2D visibility maps in a median-balanced *kd*-tree for efficient range search. We call this structure the *visibility cache* as the concept is similar to (ir)radiance caching [WRC88, KGPB05].

Each visibility map is called a *cache record*, and encodes the visibility over the sphere in world space. Essentially, these maps are just low-resolution black and white images representing the visibility of the background. A weighted average of visibility maps is multiplied by the environment map and the reflectance at each pixel. This gives a good approximation of the direct illumination, which is used as a control variate term.

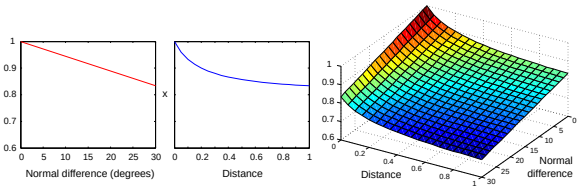


Figure 5: *The weighting function used in cache lookups.*

There are several advantages with our representation over working in the local surface frame. First, we need to quickly integrate our cache records against $L\tilde{B}$ obtained by the importance sampling algorithms [CETC06, CAM08], which operate in world space. Second, interpolation becomes trivial, as we do not have to perform rotations. The approximated visibility at a point, $\mathbf{x}$, is obtained as a linear combination of n nearby cache records:

$$\tilde{V}(\mathbf{x},\omega) = \sum_{i=1}^{n} \beta_i \tilde{V}(\mathbf{x}_i,\omega) = \sum \beta_i \tilde{V}_i. \qquad (8)$$

As a first step, we send a small number, N_{startup}, of rays (e.g., 1000) uniformly distributed over the image, and insert a cache record at the first hit along each ray. This bootstrapping of the cache completes very quickly and is only done for the first frame in an animation. Otherwise, the algorithm is a one-pass method, and new records are inserted along the way. Next, we will describe how the weights, β_i, are found.

5.1. Cache Lookups

In order to find a small set of n (typically 3–4) cache records that are representative for the visibility at $\mathbf{x}$, we start by performing a range search in the *kd*-tree. The search is restricted to the m (typically 10–20) nearest records that are within a distance $d_{\max}$, and that have a similar surface orientation, i.e., $\theta_i < \theta_{\max}$, where θ_i denotes the angle between the normals at $\mathbf{x}$ and $\mathbf{x}_i$. We compute a weight, $w_i = w_g \cdot \hat{w}(d,\theta)$, for each record and then pick the n records that score the highest. These weights are then used as β_i after normalization so that $\sum \beta_i = 1$.

The purpose of $\hat{w}$ is essentially to relate d to θ in a sensible way. In our measurements, we have seen that the correlation often falls off near linearly with increasing θ, while there is a quicker dropoff in the beginning with increasing distance, d, and slower at the end. Figure 3 shows a typical example of this. We have designed a function w, which mimics this behavior (see Figure 5), and is given by:

$$w(d,\theta) = (1-\theta/\pi)\left(1 - \frac{x}{1+\lambda x}\right), \quad \text{where } x = \frac{d}{d_{\max}}, \qquad (9)$$

and λ controls how steep the initial dropoff is (we use $\lambda = 5$). Equation 9 gives a weight in the range $[\underline{w}, 1]$, where the lower limit, $\underline{w}$, can easily be derived from $d_{\max}$ and $\theta_{\max}$. Before use, we normalize w to the unit interval by setting $\hat{w} = (w - \underline{w})/(1-\underline{w})$.

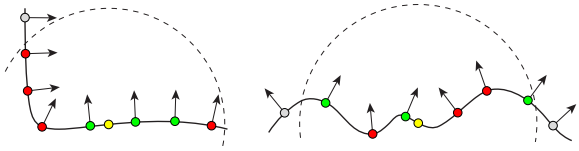


Figure 6: *To create a visibility approximation at the yellow points, we locate the m nearest cache records, and then select only the ones with similar normal directions (green).*

The motivation for first looking at a larger neighborhood, and then picking a smaller set of records, is that we want to find records with matching surface orientation. Statistically seen, these should be good approximations to V, at least if they are reasonably close. In Figure 6, we illustrate a case where a larger search region is beneficial. On a rough surface, e.g., using displacement mapping, our method only includes records with similar normals, while ignoring others, even if they are closer. To keep the cache density approximately constant in screen space, we also compute the area of the current pixel projected onto the plane passing through $\mathbf{x}$ with normal n. The maximum search range, $d_{\max}$, is then set to a fixed constant times the square root of the projected pixel area [TL04].

Finally, we include a geometric term, w_g, to reduce the weights of records that lie "in front" or "behind" the current point $\mathbf{x}$. Consider the two cases shown in Figure 7. It is desirable to reduce the weight for the record at $\mathbf{x}_i$ in the rightmost case, as part of the hemisphere at $\mathbf{x}$ must be occluded by the (unknown) geometry holding $\mathbf{x}_i$. The same holds true for the inverse case; if $\mathbf{x}$ is in front of $\mathbf{x}_i$, part of $\mathbf{x}_i$'s hemisphere must be occluded. The angle, ϑ, between the normal and the vector, v, pointing toward the record, gives an indication of the induced error. We use the simple formula $w_g = \sqrt{1-|n\cdot v|}$ as an approximation.

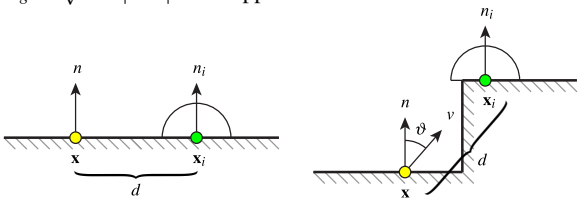


Figure 7: *If we only consider the distance and normals, the weight for the record at $\mathbf{x}_i$ would be the same in both these cases, as the distance, d, and the normals, n and n_i, are the same. However, the information stored at $\mathbf{x}_i$ is clearly less relevant at $\mathbf{x}$ for the case on the right. Thus, we include a geometric term, which takes the angle ϑ into account.*

5.2. Adaptive Refinement

During rendering, we insert a new cache record whenever the local cache density is too low. Our strategy is somewhat similar to adaptive refinement in (ir)radiance caching [KBPv06].

We first use a geometry-based criteria, inserting a new record when the weight of the highest ranked record according to Equation 9 is below a pre-determined threshold, i.e., $\max(w_i) < w_{\min}$. This will locally increase the cache density on curved surfaces.

Second, heuristics are used to decide whether to use the existing records or insert a new. We measure the *average difference* between the n nearby records found as described in the previous section. The difference, δ_{ij}, between two visibility approximations, $\tilde{V}_i$ and $\tilde{V}_j$, is defined as the probability of a random sample returning different values. As $\tilde{V}$ are discrete visibility maps, this is simply the number of pixels with different values in $\tilde{V}_i$ and $\tilde{V}_j$, divided by the total number of pixels. The average difference, $\bar{\delta}$, is the mean over all combinations of the n records, which is fast to compute as n is very small. If the difference is above a certain threshold, i.e., $\bar{\delta} > \bar{\delta}_{\max}$, we insert a new cache record at the current position, $\mathbf{x}$.

As we explicitly compare the stored visibility information, our method automatically inserts new cache records in regions of difficult occlusion. This happens even if the surface, on which the records themselves are placed, is simple. Records near an occluder will "see" different things, and hence the cache will be locally refined until the difference is below the threshold. The results of our two refinement criteria, the geometry-based and the average visibility difference, are visualized in Figure 8.

The insertion of a new record in the *kd*-tree is done by adding it to the leaf node enclosing its position. If a leaf gets too large (e.g., more than four records), we split it along its median. As each insertion makes the tree progressively more unbalanced, we occasionally rebalance the whole tree. As the number of stored records is rather small (in the order of 10,000), this only takes a few milliseconds. Note that we use a standard 3D *kd*-tree based on position only. To further speed up the proximity search, it would be possible to use a higher-dimensional tree, splitting on both position and normal orientation. Our rendering application is multi-threaded, so we also protect all accesses to the *kd*-tree with a read-write lock. This way, multiple threads may read simultaneously, but only one at the time can write.

5.3. Exploiting Temporal Coherence

A major advantage of the visibility cache is that we can reuse our world-space cache records over time, thereby reducing the computations performed per frame. In many cases, changes in visibility are local and do not affect the whole scene, and for the case of just a moving camera, the visibility field does not change at all. As we use the visibility approximations as control variates, reusing slightly inaccurate visibility information will *not* introduce bias or artifacts, only increase the noise.

To avoid the visibility information from deteriorating, we automatically remove inaccurate cache records. In computing the outgoing radiance using Equation 2, we have to evaluate both the exact visibility, V, and the approximation, $\tilde{V}$, for each of the sampling directions obtained by importance sampling. We count the number of rays for which the visibility approximation is off, and accumulate this value in the cache records. At the end of each frame, we remove all records above a predefined threshold, $\varepsilon_{\max}$, e.g., 5% misses. Another method is to sort the records according to their error rate and remove the k worst performing. To keep the cache small, we also remove all records that have not been used for a certain number (e.g., 10) of frames.

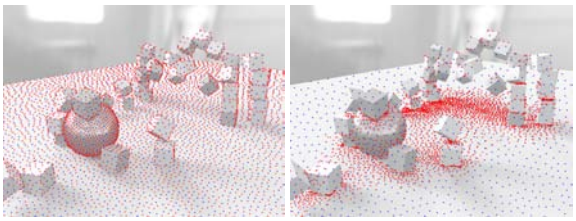


Figure 8: *The red dots show the locations of cache records. The geometry-based criteria (left) puts more records on surfaces of high curvature, while the heuristic based on measuring the average visibility difference (right) focuses on regions of difficult occlusion. Note that this method efficiently finds the regions of low correlation (see Figure 4).*

5.4. Setting the Thresholds

The different parameters controlling the behavior of the visibility cache can be classified based on which feature they control, as shown in Table 1. This makes their setup more intuitive. Some of the parameters are non-critical and reasonable default values work well, e.g., $N_{\text{startup}} = 1000$ and $\theta_{\max} = 30°$. Others can be derived automatically, e.g., $d_{\max} =$ 5% of the image width in pixels. The thresholds controlling the insertion ($w_{\min}$ and $\bar{\delta}_{\max}$) and removal ($\varepsilon_{\max}$) of cache records have a direct impact on the size of the cache, and have to be manually set.

Category	*Parameter*	*Description*
initialization	N_{startup}	Number of initial records
search	$d_{\max}$	Max search distance
	$\theta_{\max}$	Max normal difference
insertion	$w_{\min}$	Minimum weight allowed
	$\bar{\delta}_{\max}$	Max average visibility difference between nearby records
removal	$\varepsilon_{\max}$	Max number of wrong visibility queries

Table 1: *The parameters controlling the visibility cache.*

In general, we have seen that the cache adapts well to different scenes. With the same parameters, a scene with complex occlusion will get more cache records than a scene with

simple geometry. This also means that for a dynamic animation sequence, more records will be allocated to difficult frames. We have found it useful to have a couple of predefined setups (i.e., low/medium/high scene complexity), and then manually adjust the threshold values only if needed.

It would be interesting to look into ways of automatically determining good starting values for the different parameters. We could perhaps use statistics gathered during the initial bootstrapping phase to estimate the scene complexity. Some of the work by Feixas et al. on analyzing scene complexity [FdABS99] could potentially be used for this purpose. This has been saved for future work.

5.5. Implementation

To create a visibility map, we divide the domain into $2^M \times 2^M$ pixels, and evaluate the visibility through ray tracing with one ray per pixel. The sample locations are stratified within the cells, and to improve the blue-noise characteristics of the sampling pattern, we perform a small number (10–30) of iterations of Lloyd-relaxation. We use an area-preserving mapping [CAM08] of the sphere, which is based on the octahedral map.

Since V is a binary function, we have opted for a compact *bitwise* representation of the cache records. In total we need 2^{2M} bits to store an uncompressed visibility map. The bits are encoded using the Z-order (Morton-order) space-filling curve. The index, z, of a two-dimensional coordinate, (x,y), is found by bitwise interleaving the binary representations of its coordinates, $x = (x_{M-1} \ldots x_0)_2$ and $y = (y_{M-1} \ldots y_0)_2$ respectively, as follows:

$$z = (y_{M-1}x_{M-1} \ldots y_1x_1y_0x_0)_2. \qquad (10)$$

Due to its locality-preserving behavior, this encoding implicitly provides a quadtree representation of the visibility. The bits representing a node are always consecutive, so the visibility can be found by simple bit shifts and logical operations, e.g., if a node contains all zeroes, it is fully occluded. At higher levels, we use a secondary hierarchy with 2-bit values indicating full/partial/no visibility. See Figure 9. Another advantage of using a bitwise representation is that we can very quickly find the mean difference, $\delta_{ij} = \sum |\tilde{V}_i - \tilde{V}_j| / 2^{2M}$, between two visibility maps. We compute δ_{ij} as follows:

$$\delta_{ij} = \frac{\#\text{nonzero bits in } \tilde{V}_i \oplus \tilde{V}_j}{2^{2M}}, \qquad 0 \le \delta_{ij} \le 1, \qquad (11)$$

i.e., we *xor* the binary visibility representations and count the number of nonzero bits in the result. This can be done very efficiently using SIMD-optimized code.

To quickly find $J = \int L\tilde{B}\tilde{V}\,d\omega$ (see Equation 2), which serves as an approximation to the sought-after integral, we traverse the quadtree representation of $L\tilde{B}$ [CAM08], starting at its root. For each node, we look at the visibility, and stop the recursion if the node is occluded. If it is fully visible, we add up the integral of $L\tilde{B}$ over the node, which is already computed in the importance sampling step, and if the node is only partially visible, we recursively traverse its four children.

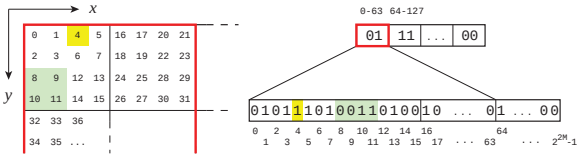


Figure 9: *We encode the binary visibility map as a two-level Z-order hierarchy, which gives an implicit quadtree representation.*

6. Other Applications

The visibility cache is an adaptively and sparsely sampled representation of the 4D visibility field. Our prime application is Monte Carlo rendering. However, the same framework can be used in a number of other applications.

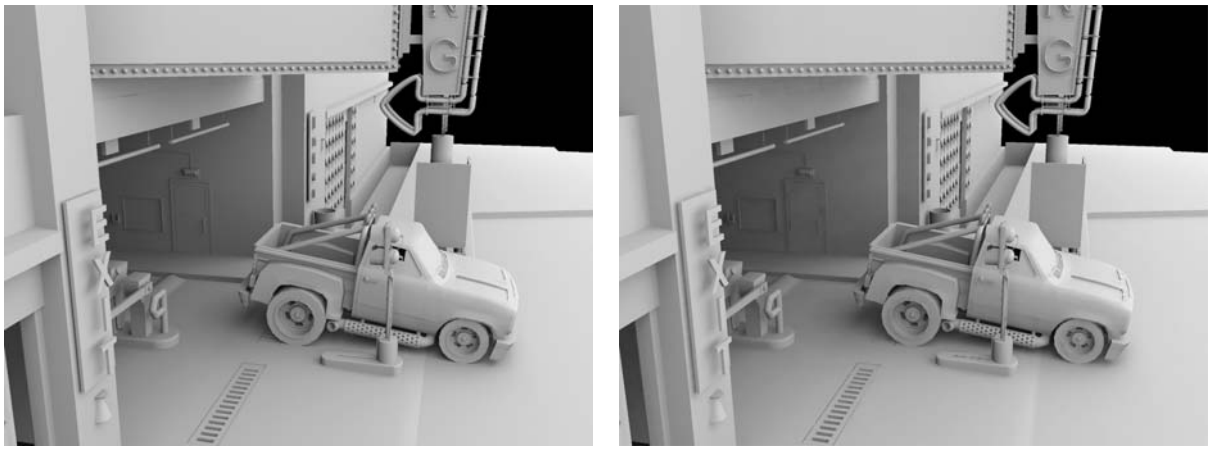
6.1. Ambient Occlusion

Ambient occlusion [ZIK98] is a widely used technique for adding realism to local shading models. The ambient occlusion term, A, is the integral of the visibility function over the hemisphere, taking solid angle into account, as follows:

$$A(\mathbf{x}) = \int_\Omega V(\mathbf{x},\omega)(n\cdot\omega)\,d\omega. \qquad (12)$$

This can be evaluated using ray tracing, but for high-quality results without banding, up to 1000 rays/pixel are needed. By replacing V by our visibility approximation, $\tilde{V}$, obtained from the cache, we get a quick approximation of A. For example, for full HD rendering (resolution 1920×1080) and a visibility cache with 20k records of resolution 32×32, the amortized cost is only 5 rays/pixel. Since the cache is adaptively refined, the method handles regions of difficult occlusion very well.

We compute the ambient occlusion term, A, during the creation of each new visibility record, and store the value in the cache record. For shading a pixel, we locate the m nearest records and compute a weighted average of their preintegrated A values, rather than first selecting a smaller set of records as before. To get a smoother solution, we use a Gaussian filter, whose width is set based on the projected pixel area. We also compute the distance, R_i, to the nearest intersection at each cache record in order to better detect small geometric features, similar to [TL04]. If $\max[w_i \cdot (1 - d/R_i)] < w_{\min}$, then a new record is inserted. Figure 10 shows the result for a typical scene containing about 259k triangles. A major strength of our approach is that the solution is noise-free, although a few artifacts due to the sparse sampling exist.



Reference (2776 seconds) Visibility Cache (44 seconds)

Figure 10: *The left image shows a ray traced ambient occlusion reference image using 4 samples/pixel and 1024 rays per shading point. For the right image, we have used our visibility cache for evaluating the ambient occlusion term, using on average 17.7 rays per shading point. The speedup is a factor 63× for this scene. However, some artifacts are visible, e.g., around the door, and the overall appearance is a bit softer.*

6.2. Lighting Design

The integral, J, over the control variate term (see Equation 2), is an approximation of the outgoing radiance based on the exact lighting and approximations of the reflectance and visibility, $\tilde{B}$ and $\tilde{V}$ respectively. By directly visualizing J, we get a very quick *preview* of the direct illumination. This can be useful for, e.g., fine-tuning the lighting in a scene before starting a production-quality rendering.

Figure 11 shows a preview image computed using 12,500 cache records. The output is fairly blotchy, but the quality is good enough to judge where shadows and highlights fall. More advanced interpolation strategies should improve the quality, and we plan to develop this idea further. The main execution cost currently lies in computing $\tilde{B}$ per pixel. A system for caching and interpolating $\tilde{B}$ (i.e., a shader cache), similar to our visibility cache, would be one way of decreasing the cost.

Reference Preview (12,500 cache records)

Figure 11: *Direct visualization of the control variate term gives a quick preview without doing any per-pixel sampling. After the BRDF quadtrees have been setup, the environment map can be replaced and/or rotated freely, providing an almost instant preview of the shading, including glossy effects.*

7. Results

We have implemented our algorithm for direct illumination on top of a recent technique for product importance sampling [CAM08], which samples the product of a distant area light source (e.g., an environment map) and the local BRDF. All parts of their algorithm are carried out unchanged. The only difference is that, for each pixel, we perform a lookup in our visibility cache, and evaluate the obtained visibility approximation for each of the sampling directions. These visibility approximations are subtracted from the estimator, and finally the integral of the triple product, $J = \int L\tilde{B}\tilde{V}$, is computed and added to the result, as described by Equation 3. All results were generated on a Mac Pro with an Intel "Penryn" 45nm processor at 3.2GHz, using 2 cores.

Scene	**garage**	**dynamics**	**garden1**
#Records	14,300	6,200	17,100
Record size	32×32	64×64	32×32
Avg rays/pixel	3.81	6.61	4.56

Table 2: *The number of cache records and resolutions used for three of our test scenes. The last row shows the amortized cost of the visibility cache, measured as the average number of shadow rays/pixel at 1600×1200 pixels resolution.*

The performance was evaluated on three scenes of different complexity; *garden1* and *dynamics* from Figure 4, and the *garage* scene in Figure 11. For most tests we have used a visibility map resolution of 32×32 pixels. In simple scenes where ray tracing is fast, a higher resolution of 64×64 pixels can be used, but the extra cost is usually not motivated by a large enough quality improvement. The settings used for our three test scenes are summarized in Table 2, and the

	garage				dynamics				garden1			
#Samples	10	30	100	300	10	30	100	300	10	30	100	300
Time [CAM08]	103.0	155.3	325.1	779.1	65.0	82.1	135.2	280.5	229.1	451.8	1211.7	3220.6
Time [our] (s)	124.9	178.3	346.8	800.9	85.0	102.6	157.9	303.4	309.2	534.7	1307.1	3306.8
Overhead (s)	21.9	23.0	21.7	21.8	20.0	20.5	22.7	22.9	80.1	82.9	95.4	86.2
Overhead (%)	21.3%	14.8%	6.7%	2.8%	30.8%	24.9%	16.8%	8.1%	34.9%	18.3%	7.9%	2.7%
Variance ratio	0.212	0.339	0.497	0.633	0.405	0.533	0.641	0.734	0.775	0.997	1.190	1.244

Table 3: *Statistics for three of our scenes rendered at 1600×1200 pixels resolution.*

rendering results are presented in Table 3. We present the variance reduction as the ratio of variance in the images rendered with our algorithm to the ones rendered using only importance sampling [CAM08].

For the *garage* scene there is a significant variance reduction, ranging from almost 5× at 10 samples/pixel to 37% at 300 samples/pixel. The lighting in this scene is representable for scenes where ordinary product importance sampling fails to give good results. The majority of the light is coming from a few large, bright light sources, which are occluded by the building. Hence, most rays are occluded, and there is strong noise in the shadow regions. This is similar to what happens in Figure 1. In these types of scenes, our algorithm gives a large improvement at a modest cost. Figure 13 and 14 show a comparison between our method and that of Clarberg et al. [CAM08]. At equal rendering time (178 s), the variance is reduced to less than half (variance ratio 0.489).

The *dynamics* scene is a hard case as it has very little occlusion. The control variate term gives a noise reduction also in unoccluded regions, but we have found the effect to be much stronger in shadows. However, our algorithm still gives a 37–60% reduction of variance, at a rendering time overhead of only 8% to 31%. There is a net win, but the improvement is not as large as we had hoped for.

Finally, the *garden1* scene presents a worst-case scenario with near-random visibility. Our analysis in Section 4 shows that there is a rather weak correlation in the visibility function. In order to exploit this, a large number of cache records would be needed. Using a reasonable number of records (17,100), we achieve only a very modest variance reduction at 10 and 30 samples/pixel. At the higher sampling rates, there is actually an *increase* in variance. This may seem counterintuitive, since Equation 5 states that the variance can only decrease with control variates. However, that is under the assumption that the optimal value of α is known (Section 3), which is not the case.

The memory overhead of our visibility cache is very modest. Each cache record of resolution 32×32 occupies 172 bytes (including additional book-keeping data), which means a cache with 14,300 records (as we used for the garage scene) uses 2.4 MB memory. This fits well within the L2 cache on modern CPUs. In general, we have found the algorithm to complement existing methods for product sampling very well. A strong feature of our algorithm is that

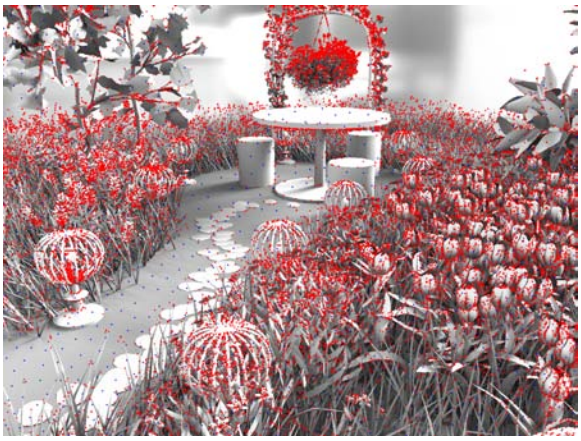


Figure 12: *The distribution of cache records for the garden1 scene, which presents a difficult case for our algorithm.*

it provides a more robust solution than product sampling alone, as illustrated by Figure 1. The largest quality improvement is achieved in scenes with heavy occlusion, and especially in scenes with bright light sources that are occluded. Both these cases are difficult for product importance sampling. For more results, we refer to the supplemental video.

8. Limitations and Discussion

We have restricted the analysis and algorithm to *binary* visibility. This makes an efficient bitwise implementation possible, but it also implies that only non-local lighting can be used, i.e., light sources that lie outside the convex hull of the scene. Our algorithm is, in theory, not limited to only distant lighting, but it makes most sense when combined with recent methods for importance sampling under environment map illumination [CETC06, CAM08]. These algorithms exploit the fact that the lighting, L, does not change per pixel.

By storing the distance to the nearest occluder rather than just a binary value, we could adapt our method to handle local lighting. This would require substantial changes to the implementation, and the size of the visibility records would grow considerably, making its usefulness questionable. It would, however, be interesting to perform a similar analysis as in Section 4 to the case of local visibility. For binary visibility, nearby points and points with similar normals have highly correlated visibility. We expect the same

to be true for local visibility, but new questions arise, e.g., how occluder distance correlates with normal/positional differences. It would also be interesting to perform a statistical analysis of the other terms in the rendering equation. This could be useful for improving, e.g., indirect illumination.

Implementation-wise, there are a couple of issues we would like to address. The direct visualization of visibility in Section 6 reveals that the interpolation is far from perfect. With better weights, we believe most of the artifacts in the ambient occlusion and lighting design examples can be removed. This would also further reduce the noise in MC rendering. An approach similar to irradiance gradients [WH92] should be possible. There are also a number of optimization to do, e.g., caching the visibility difference between nearby records instead of recomputing them for each lookup, using SIMD in the integration of the triple product, and so on.

9. Conclusion

The contribution of this paper is twofold. First, we study the statistical properties of the visibility function. Our insights here can be useful when designing algorithms taking visibility into account. We believe this is the next logical step in photo-realistic rendering, as many existing algorithms only consider the product of lighting and reflectance.

Second, we propose to use control variates to incorporate a visibility approximation in MC rendering. The key idea is to evaluate the difference between the exact function and the estimation from our visibility cache. The method is attractive in that it can exploit both spatial and temporal coherence, without introducing bias. We believe the same concept can be applied to other applications, e.g., radiance caching.

Acknowledgements

The garage scene was modeled by Christophe Desse and Matthew Thain, and the dynamics scene was created by Jacob Munkberg. We would also like to thank all the anonymous reviewers. This work was supported by the Swedish Foundation for Strategic Research and by Intel Corporation.

References

[ARHM00] AGRAWALA M., RAMAMOORTHI R., HEIRICH A., MOLL L.: Efficient Image-Based Methods for Rendering Soft Shadows. In *Proceedings of ACM SIGGRAPH 2000* (2000), pp. 375–384.

[BARA06] BEN-ARTZI A., RAMAMOORTHI R., AGRAWALA M.: Efficient Shadows from Sampled Environment Maps. *Journal of Graphics Tools, 11*, 1 (2006), 13–36.

[BGH05] BURKE D., GHOSH A., HEIDRICH W.: Bidirectional Importance Sampling for Direct Illumination. In *Eurographics Symposium on Rendering* (2005), pp. 147–156.

[CAM08] CLARBERG P., AKENINE-MÖLLER T.: Practical Product Importance Sampling for Direct Illumination. *Computer Graphics Forum (Proceedings of Eurographics 2008), 27*, 2 (2008), 681–690.

[CETC06] CLINE D., EGBERT P. K., TALBOT J. F., CARDON D. L.: Two Stage Importance Sampling for Direct Lighting. In *Eurographics Symposium on Rendering* (2006), pp. 103–113.

[CJAMJ05] CLARBERG P., JAROSZ W., AKENINE-MÖLLER T., JENSEN H. W.: Wavelet Importance Sampling: Efficiently Evaluating Products of Complex Functions. *ACM Transactions on Graphics, 24*, 3 (2005), 1166–1175.

[DWB*06] DONIKIAN M., WALTER B., BALA K., FERNANDEZ S., GREENBERG D. P.: Accurate Direct Illumination Using Iterative Adaptive Sampling. *IEEE Transactions on Visualization and Computer Graphics, 12*, 3 (2006), 353–364.

[FBG02] FERNANDEZ S., BALA K., GREENBERG D. P.: Local Illumination Environments for Direct Lighting Acceleration. In *Eurographics Workshop on Rendering* (2002), pp. 7–14.

[FCH*06] FAN S., CHENNEY S., HU B., TSUI K.-W., LAI Y.-C.: Optimizing Control Variate Estimators for Rendering. *Computer Graphics Forum (Proceedings of Eurographics 2006), 25*, 3 (2006), 351–357.

[FdABS99] FEIXAS M., DEL ACEBO E., BEKAERT P., SBERT M.: An Information Theory Framework for the Analysis of Scene Complexity. *Computer Graphics Forum (Proceedings of Eurographics 1999), 18*, 3 (1999), 95–106.

[GBP07] GAUTRON P., BOUATOUCH K., PATTANAIK S.: Temporal Radiance Caching. *IEEE Transactions on Visualization and Computer Graphics, 13*, 5 (2007), 891–901.

[GH06a] GHOSH A., HEIDRICH W.: Correlated Visibility Sampling for Direct Illumination. *The Visual Computer, 22*, 9 (2006), 693–701.

[GH06b] GHOSH A., HEIDRICH W.: Sequential Sampling for Dynamic Environment Map Illumination. In *Eurographics Symposium on Rendering* (2006), pp. 115–126.

[HDG99] HART D., DUTRÉ P., GREENBERG D. P.: Direct Illumination with Lazy Visibility Evaluation. In *Proceedings of ACM SIGGRAPH 99* (1999), pp. 147–154.

[Jen01] JENSEN H. W.: *Realistic Image Synthesis Using Photon Mapping*. A K Peters, 2001.

[Kaj86] KAJIYA J. T.: The Rendering Equation. *Computer Graphics (Proceedings of ACM SIGGRAPH 86), 20*, 4 (1986), 143–150.

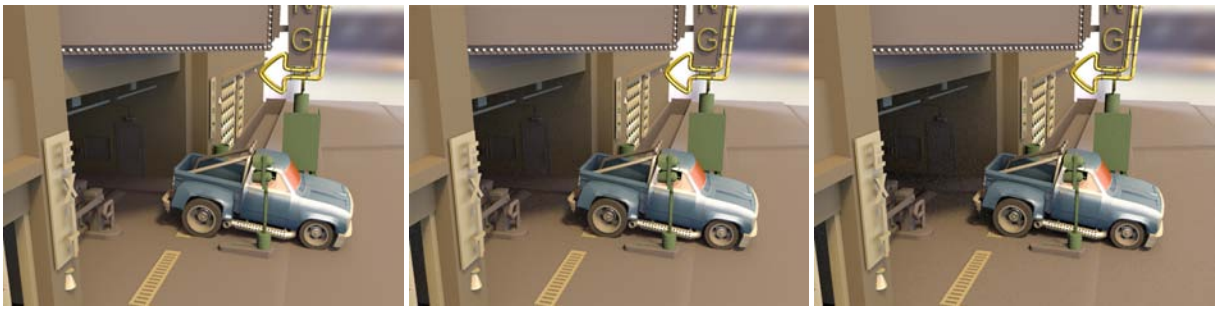
[KBPv06] KŘIVÁNEK J., BOUATOUCH K., PATTANAIK S. N., ŽÁRA J.: Making Radiance and Irradiance Caching Practical: Adaptive Caching and Neighbor Clamping. In *Eurographics Symposium on Rendering* (2006), pp. 127–138.

[KGPB05] KŘIVÁNEK J., GAUTRON P., PATTANAIK S., BOUATOUCH K.: Radiance Caching for Efficient Global Illumination Computation. *IEEE Transactions on Visualization and Computer Graphics, 11*, 5 (2005), 550–561.

[KW86] KALOS M. H., WHITLOCK P. A.: *Monte Carlo Methods*. John Wiley & Sons, 1986.

[LW94] LAFORTUNE E. P., WILLEMS Y. D.: The Ambient Term as a Variance Reducing Technique for Monte Carlo Ray Tracing. In *Eurographics Workshop on Rendering* (1994), pp. 168–176.

[LW95] LAFORTUNE E. P., WILLEMS Y. D.: A 5D Tree to Reduce the Variance of Monte Carlo Ray Tracing. In *Eurographics Workshop on Rendering* (1995), pp. 11–20.



Reference Our algorithm Clarberg et al. [CAM08]

Figure 13: *Equal-time comparison between our algorithm and that of Clarberg et al. [CAM08], using 30 and 39 samples/pixel respectively. The rendering time is 178 s at resolution 1600×1200, and the variance is reduced by 51.4%, i.e., to less than half.*

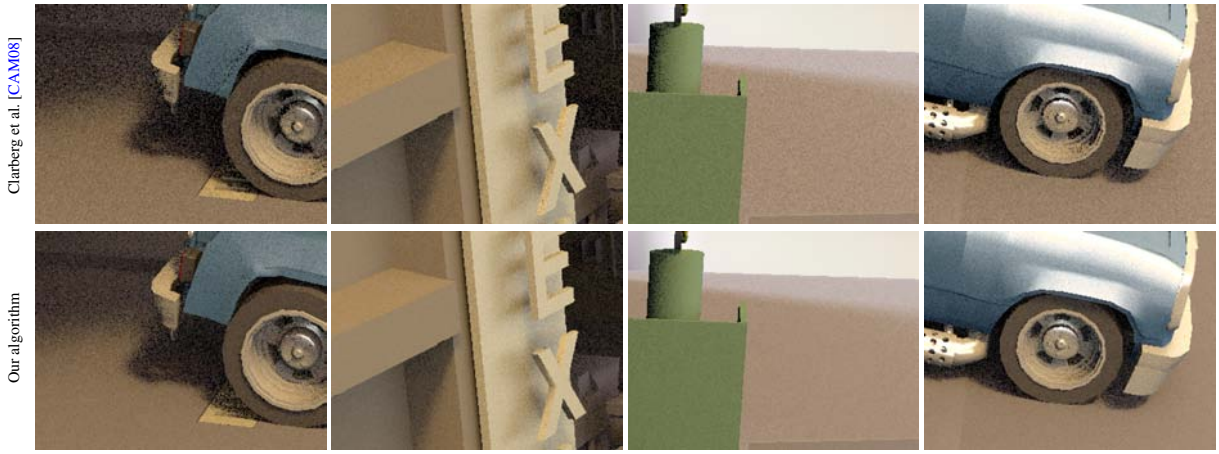


Figure 14: *Equal-time comparison with crops from the full resolution images in Figure 13.*

[MA06] MEYER M., ANDERSON J.: Statistical Acceleration for Animated Global Illumination. *ACM Transactions on Graphics, 22*, 3 (2006), 1075–1080.

[SKCA01] SZIRMAY-KALOS L., CSONKA F., ANTAL G.: Global Illumination as a Combination of Continuous Random Walk and Finite-Element Based Iteration. *Computer Graphics Forum (Proceedings of Eurographics 2001), 20*, 3 (2001), 288–298.

[SSSK04] SZÉCSI L., SBERT M., SZIRMAY-KALOS L.: Combined Correlated and Importance Sampling in Direct Light Source Computation and Environment Mapping. *Computer Graphics Forum (Proceedings of Eurographics 2004), 23*, 3 (2004), 585–594.

[SWZ96] SHIRLEY P., WANG C., ZIMMERMAN K.: Monte Carlo Techniques for Direct Lighting Calculations. *ACM Transactions on Graphics, 15*, 1 (1996), 1–36.

[TCE05] TALBOT J., CLINE D., EGBERT P.: Importance Resampling for Global Illumination. In *Eurographics Symposium on Rendering* (2005), pp. 139–146.

[TL04] TABELLION E., LAMORLETTE A.: An Approximate Global Illumination System for Computer Generated Films. *ACM Transactions on Graphics, 23*, 3 (2004), 469–476.

[VG95] VEACH E., GUIBAS L. J.: Optimally Combining Sampling Techniques for Monte Carlo Rendering. In *Proceedings of ACM SIGGRAPH 95* (1995), pp. 419–428.

[WABG06] WALTER B., ARBREE A., BALA K., GREENBERG D. P.: Multidimensional Lightcuts. *ACM Transactions on Graphics, 25*, 3 (2006), 1081–1088.

[War91] WARD G.: Adaptive Shadow Testing for Ray Tracing. In *Eurographics Workshop on Rendering* (1991), pp. 11–20.

[WFA*05] WALTER B., FERNANDEZ S., ARBREE A., BALA K., DONIKIAN M., GREENBERG D. P.: Lightcuts: A Scalable Approach to Illumination. *ACM Transactions on Graphics, 24*, 3 (2005), 1098–1107.

[WH92] WARD G. J., HECKBERT P.: Irradiance Gradients. In *Eurographics Workshop on Rendering* (1992), pp. 85–98.

[WRC88] WARD G. J., RUBINSTEIN F. M., CLEAR R. D.: A Ray Tracing Solution for Diffuse Interreflection. *Computer Graphics (Proceedings of ACM SIGGRAPH 88), 22*, 4 (1988), 85–92.

[ZIK98] ZHUKOV S., IONES A., KRONIN G.: An Ambient Light Illumination Model. In *Eurographics Workshop on Rendering* (1998), pp. 45–55.