RH-TR-2003-1: Estimating reflected radiance under complex distant illumination

Jonathan M. Cohen*
Rhythm and Hues Studios

Abstract

We present an unbiased Monte Carlo technique for estimating the value of reflected radiance at a surface point due to a hemisphere of direct distant illumination. We use an importance sampling estimator with a novel piecewise-constant importance function that effectively concentrates ray samples where energy is likely to be found. The importance function is efficient to evaluate and draw samples from, and is chosen to minimize its squared distance from the integrand of the radiance integral, even though the exact form of this integrand is unknown.

To properly account for the effects of the visibility term in the shading calculation, we propose the use of a *shadow cache* which caches information about which ray directions are occluded or unoccluded from a point in space. We can incorporate this information into the importance function to automatically concentrate hemispherical samples where the light source is likely to be unoccluded, thereby increasing the efficiency of the estimator.

We present visual and numerical results that demonstrate the new estimator can give orders of magnitude lower error than simpler sampling techniques for highly complex lighting situations.

CR Categories: G.3 [Mathematics of Computing]: Probability and Statistics—Monte Carlo Algorithms I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism—Color, shading, shadowing, and texture

Keywords: global illumination, Monte Carlo integration, importance sampling, shadow cache, image-based lighting

1 Introduction

A common rendering task in visual effects is to integrate computer-generated elements with a live action background. The CG models range from organic characters animated via complex physical models to static objects like buildings or vehicles. Often, all of the light in the scene emanates from outside the object, and the object passively reflects incident radiance toward the camera. In practice, these renderings are frequently achieved using a number of heuristic techniques, mostly requiring labor intensive setup and tweaking by digital artists.

As computers get faster and global illumination algorithms improve, there is an increasing desire to efficiently calculate such "outside-in" lighting situations using global illumination. A fundamental step in many of these algorithms is to compute, at a surface point, the radiance reflected toward some direction due to direct illumination (i.e., photons that have traveled directly from the light source to the surface, without having been reflected in between). We present a technique to compute this quantity efficiently in the outside-in case where the incident light has the high variation found in real-world lighting situations such as outdoors or on a cinematic set.

*e-mail: jcohen@rhythm.com

1.1 Reflected radiance

Consider an area light source that subtends solid angle Ω whose radiance output at a particular wavelength is characterized by the function $L(p,\omega)$, a function of a point in space p and a direction ω. For a particular surface point p with normal n, and ray to the virtual camera e (the "eye" ray), the reflected radiance from p along e due to the light source can be expressed as the integral

$$I(p,n,e) = \int_{\omega\in\Omega} L(p,\omega)\, f_r(\omega,n,e) V_p(\omega)(n\cdot\omega) d\omega \qquad (1)$$

where f_r is the bi-directional reflectance distribution function (BRDF) that describes how light is scattered from direction ω to e with surface normal n, and V_p is the visibility function whose value is 0 if p is occluded from the light source in direction ω and 1 otherwise.[1]

For simplicity, we can refer to the integrand as $H_{p,n,e}$ and rewrite the equation as

$$I(p,n,e) = \int_{\omega\in\Omega} H_{p,n,e}(\omega) d\omega.$$

We call H the *hemispherical transfer* function, and its SI units are $Watt/m^2/sr^2$, or radiance per solid angle. For notation's sake, we will drop the p,n,e subscripts, but it is important to remember that H varies with p, n, and e[2].

In the technique of image-based lighting (IBL), we consider a light source positioned at the sphere at infinity. Then $L(p,\omega)$ depends only on direction ω, so we can drop the p parameter. Typically, this light source is derived from measured data captured with a panoramic camera or radiance sensor, and the values of $L(\omega)$ are stored in an environment map. For the remainder of this paper, we will use L in this image-based lighting sense. Furthermore, Integral 1 is evaluated over the entire hemisphere centered around the surface normal n, which we will denote by Ω_n. We drop the n subscript when it can be inferred from the context.

The overriding cost of evaluating Equation 1 is testing the visibility term, which typically requires casting a ray into the scene. Our goal, therefore, is to minimize the number of times the visibility function must be tested by minimizing the number of hemispherical samples required to estimate Equation 1 with low noise and high accuracy. While it may be possible to use finite quadrature rules to estimate integrals such as we are interested in, Monte Carlo (MC) and Quasi-Monte Carlo (QMC) methods are the obvious choice because of their accuracy and flexibility. The environment-based importance sampling scheme presented in Section 4 produces good results with as few as 128 hemispherical samples, even in a highly dynamic lighting environment.

We focus on diffuse reflectance models because they are the hardest to sample in the case we are interested in. Because a diffuse

[1] Most authors use L_r to denote reflected radiance. We prefer the functional notation $I(p,n,e)$ in this context because it stresses that reflected radiance varies over the image plane and is a function of p, n, and e.

[2] In the case of a Lambertian BRDF, H is view-independent and hence does not not depend on e.

BRDF distribution essentially has no tail, it does not mask the extreme spikes and variations found in L or V_p. Therefore we need to be sure that the sampling pattern will find all of the features in L and V_p, which can be quite difficult since there is no analytical form for either of these functions. Furthermore, a large portion of the total CPU time spent in a large production facility such as Rhythm and Hues is spent calculating precisely this quantity: reflected radiance of Lambertian surfaces under complex distant illumination.

Our technique will generalize to any reflectance model where the BRDF (multiplied by the $cos(\omega)$ factor) can be expressed as a function of a single direction, or where the BRDF can be approximated as the product of several such functions [Kautz and McCool 1999; Latta and Kolb 2002]. Potentially, this technique could cover an extremely large and useful class of refectance distributions. However, at the time of this writing, non-Lambertian reflectance models have not been implemented.

2 Related Work

The technique of image-based lighting has existed for over 20 years in the form of environment mapping and preconvolved environment mapping, techniques which do not consider visibility [Blinn and Newell 1976; Williams 1983; Miller and Hoffman 1984; Greene 1986]. The recently introduced technique of *ambient occlusion mapping* [Christensen 2002], builds on this early work by approximating visibility with a single scale value. These two techniques will be described in Section 3.1. [Debevec 1998] demonstrated that IBL could be used in a global illumination setting by projecting the environment map onto geometry as an emissive texture map. Debevec and his colleagues also filled in a number of missing pieces of the IBL technique, such as how to capture radiometrically accurate information using a standard camera [Debevec and Malik 1997], and how to cast shadows from CG objects onto real ones [1998].

The sampling technique described in Section 4 is a form of importance sampling, which is an important field of study in numerical analysis. We refer the reader to a textbook such as [Evans and Swartz 2000] for a thorough description, or [Owen and Zhou 2000] for a an overview of modern importance sampling. Because of its good performance and flexibility, importance sampling has been used extensively in rendering algorithms. [Shirley et al. 1996] described techniques for efficiently sampling luminaires using importance sampling. [Veach and Guibas 1995] gave a formulation for optimally combining importance sampling techniques for sampling the reflectance distribution as well as light sources. Veach and Guibas later applied a Monte Carlo Markov Chain algorithm to solve the path tracing form of global illumination [Veach and Guibas 1997]. For an excellent overview of importance sampling and other Monte Carlo methods we recommend Veach's encyclopedic dissertation [1997].

Solving Equation 1 under direct illumination is a sub-problem that arises frequently in global illumination. Some of the algorithms that require a solution to this sub-problem include the final gathering step of photon mapping [Jensen 2001], distributed ray tracing [Cook et al. 1984], bi-directional path tracing [Lafortune and Willems 1993], hierarchical Monte Carlo rendering [Keller 2002], and irradiance caching [Ward et al. 1988].

Perhaps the most similar work to our own is [Jensen 1995] which guides Monte Carlo path tracing with an importance function built from information in a photon map. Like Jensen, our importance function is piecewise constant, and based on a partition of the hemisphere. The advantage of Jensen's approach is that the photon map is able to account for direct and indirect illumination simultaneously, while our importance function is based only on direct illumination. In the case that most of the light energy is direct, our importance function matches the integrand more closely. Both techniques naturally complement each other, and they could be combined directly using Veach and Guibas' balance heuristic [1995], or via an extended shadow cache as discussed in the Conclusion.

3 Existing Strategies

3.1 Approximate solutions

By ignoring the visibility term V_p, we get an approximation to Equation 1 we will refer to as the *non-shadowed approximation*. In the IBL setting, when the BRDF $f_r(n \cdot \omega)$ can be parameterized by exactly one vector, there is a simple algorithm for computing the non-shadowed approximation. In the Lambertian case, for example, $f_r(\omega, n, e)(n \cdot \omega) = n \cdot \omega$, and Equation 1 simplifies to

$$I(n) \approx I_{ns}(n) = \int_{\omega \in \Omega} L(\omega)(\omega \cdot n) d\omega. \tag{2}$$

I_{ns} is a function of the surface normal n. Thus, $I_{ns}(n)$ can be precomputed for a dense but finite set of directions and stored in a texture map. Shading a surface point with normal n, involves a single texture map lookup based on the value of n. This "diffuse preconvolution" technique (shown in Figure 1(a)) was introduced by Greene [1986] and is closely related to the theory of spherical convolution, a link thoroughly explored in [Ramamoorthi and Hanrahan 2001].

The recently introduced technique of ambient occlusion mapping (Figure 1(b)) approximates the effects of shadowing in the shading calculation [Christensen 2002]. The basic *ambient occlusion approximation* to Equation 1 is

$$I(n) \approx I_{ao}(n) = \left(\frac{1}{2\pi} \int_{\omega \in \Omega} V_p(\omega) d\omega \right) \left(\int_{\omega \in \Omega} L(\omega)(\omega \cdot n) d\omega \right). \tag{3}$$

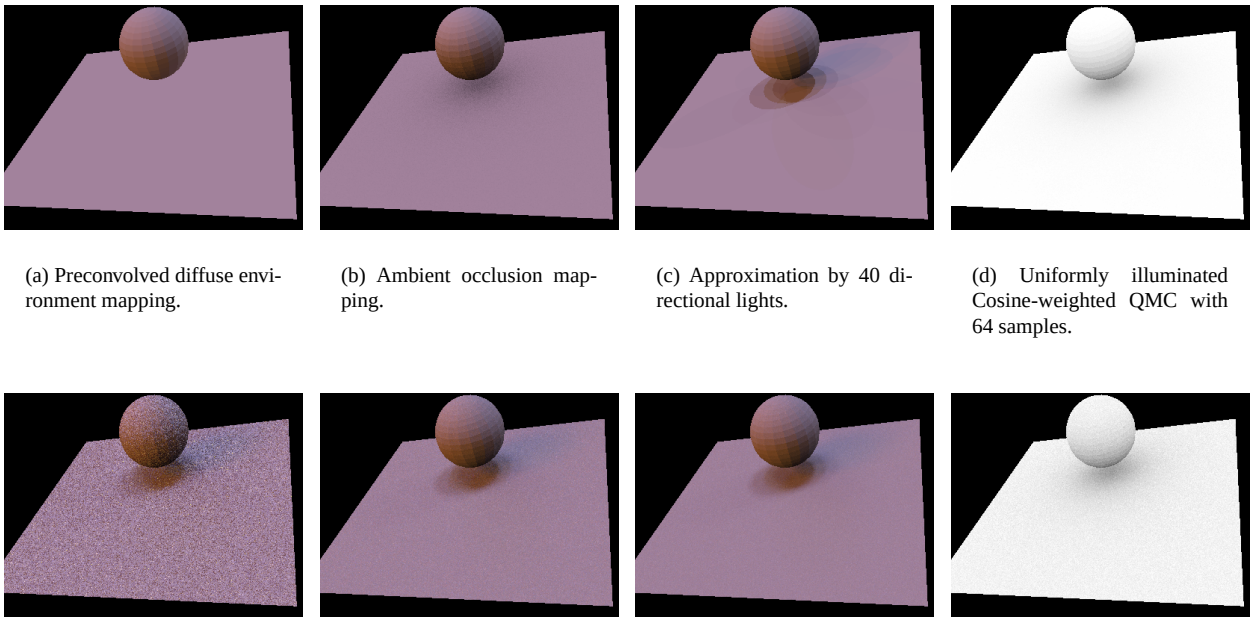
The value of the first integral is called *ambient visibility* (its complement is ambient occlusion) and may be precomputed and stored, for example, in a texture map.

There has also been work in approximations to Equation 1 using either a finite set of directional lights to approximate the continuously varying incident radiance L (shown in Figure 1(c)) [Cohen and Debevec 2001], or deterministic quadrature rules to directly estimate the integral [Kollig and Keller 2002a]. The problem with finite techniques like these is that they cannot correctly compute both soft and hard shadows without a large number of samples. Inadequate sampling resolution results in banding along the shadow regions. Kato describes a novel deterministic final gather algorithm used in the Kilauea renderer [2002] that reuses ray samples from pixel to pixel via view interpolation called "final gather reprojection." The shadow cache in Section 4.3 could be seen as a probabilistic version of Kato's approach.

Photon mapping can also be used to estimate reflected radiance. Although it is technically a biased algorithm, it is accurate enough to be considered exact if a high-enough number of photons are traced. However, the weakness of photon mapping is exactly the case we are interested in – estimating reflected radiance due to direct illumination. Although techniques such as casting shadow photons [Jensen and Christensen 1995] can estimate this quantity, Jensen [2001] recommends using path tracing to obtain the final accurate estimate in a two-pass photon mapping algorithm. Also, we are not aware of previously published techniques for efficiently incorporating image-based lighting into a photon-mapping renderer.

3.2 Exact Monte Carlo solutions

There are as many Monte Carlo techniques and ways of combining them to estimate Equation 1 as there are rendering systems. It is therefore hard to say which techniques work best, since it depend on the particular feature of H. The theory of Quasi-Monte



(a) Preconvolved diffuse environment mapping.

(b) Ambient occlusion mapping.

(c) Approximation by 40 directional lights.

(d) Uniformly illuminated Cosine-weighted QMC with 64 samples.

(e) The same as (d) under the grace environment.

(f) New sampling scheme with 64 samples.

(g) New sampling scheme with 165 samples and no visible noise.

(h) New sampling scheme under uniform lighting with 64 samples.

Figure 1: Comparison of the different techniques for estimating reflected radiance. We have intentionally rendered with no anti aliasing and used no indirect illumination in order to highlight noise and other artifacts.

Carlo integration is quite advanced, and integration schemes using randomized QMC sequences are probably the best for blind or BRDF-based sampling. For comparison purposes, we will use the randomized QMC sequence described in Section 7 of [Kollig and Keller 2002b], which is based on the Larcher-Pillichshammer radical inverse function. For uniform lighting environments, importance sampling based on the surface BRDF works very well. [Dutre 2001] describes all of the necessary formulas for Lambertian or Phong-based importance sampling. In our experience, the combination of these two techniques produces near-perfect results when the surface occlusion is fairly simple and the lighting environment is uniform or almost uniform.

Figure 1(d) shows the results of rendering a simple scene with randomized QMC cosine-weighted importance sampling under uniform lighting. This image is generated with only 64 rays per pixel, and has no visible noise. The problem with this technique, however, is that it only takes the features of f_r into account. Therefore, it will perform poorly when the features of L dominate the behavior or H, which is the case we are interested in. This can be seen in Figure 1(e), which is rendered with the same technique at the same sampling rate under the complex "grace" environment [Debevec 1998]. Figure 1(f) shows the environment-based importance sampling technique described in Section 4 at the same sampling rate. At 165 hemispherical samples, this scheme produces almost no visible noise (Figure 1(g)). A comparable noise level with the QMC integrator requires about 1000 samples.

Our technique performs slightly worse than a good BRDF-based importance sampling scheme when the environment is uniform and visibility is simple, as can be seen by comparing Figure 1(d) to Figure 1(h). This is not surprising because QMC BRDF-based schemes are close to analytically optimal in these cases and have superior stratification of samples.

4 The Sampling Scheme

```
For environment map E
  For scene geometry M
    For camera C
      For each pixel p
        For each hemispherical sample w
          Compute unoccluded contribution from
          environment map E in direction w to
          surface point of M visible at camera
          C under pixel p.
```

Figure 2: The image-based lighting process.

Figure 2 gives pseudo code for how the IBL process fits into a production workflow. To optimize the use of IBL within such a workflow, our guiding principle is to to factor as much as possible out of the inner loops. In particular, if there is a way to decrease the required number of hemispherical samples, we wouldn't mind spending several hours of computation time if this cost is incurred only once per environment map.[3] Scenes and cameras change more often, but we can still afford to spend several minutes of precomputation at the scene or camera level. We also incur some overhead per evaluation of Equation 1 (i.e. every pixel or sub-pixel) that makes our technique slower than a simpler sampling technique. However, the savings gained by decreasing the number of required hemispherical samples quickly outweigh this "startup" overhead.

[3] Actual precomputation times are about 20 minutes, depending on the quality settings.

4.1 General importance sampling

Importance sampling is an unbiased Monte Carlo technique that succeeds by placing more samples where the integrand has more energy. Given a function f over domain D, to estimate the integral

$$I = \int_{x \in D} f(x) d\mu(x) \tag{4}$$

using importance sampling, we need a function $p(x)$ such that samples can be drawn from D according to p.d.f. p, and p is easy to evaluate at any point in D. Given N samples $\{X_i\}_{i=1}^N$ drawn from D, $X_i \sim p$, the importance sample estimator of Equation 4 is

$$\hat{I}_{p,N} = \frac{1}{N} \sum_{i=1}^{N} \frac{f(X_i)}{p(X_i)}.$$

Define the residual $r(x) = f(x) - Ip(x)$. $\hat{I}_{p,N}$ is a random variable, and its variance is given by

$$Var\left[\hat{I}_{p,N}\right] = \frac{1}{N} \int_{x \in D} \frac{r(x)^2}{p(x)}. \tag{5}$$

Clearly this is minimized when $r(x)^2/p(x)$ is close to 0, which happens when $p(x) \approx f(x)/I$. We refer the reader to [Owen and Zhou 2000] for a more detailed discussion of the pitfalls and benefits of importance sampling.

Importance sampling has been used, for example, in [Veach and Guibas 1995], to evaluate reflected radiance by matching features in the BRDF f_r. This works well for uniform area light sources. To apply it most effectively in IBL, however, we would like an importance function that matches all of the features in the hemispherical transfer function $H_{p,n,e}$. This is difficult for two reasons. First, it is necessary to pick an importance function that is easy to draw samples from. We propose using a piecewise-constant approximation to H that can be rapidly computed and sampled from. Second, H is different for each point p because the visibility function V_p changes with p. We propose using a *shadow cache* to estimate the value of V_p based on nearby surface samples, which can then be combined with the piecewise-constant importance function to yield an efficient importance sampling scheme.

4.2 Environment-based importance sampling

We begin by splitting H into a sum of several functions, each with small support. Let $\{T_i\}_{i=1}^M$ be a partition of the unit hemisphere Ω_n in the sense that $T_i \cap T_j \neq \emptyset \Rightarrow i = j$ and $\cup_{i=1}^M T_i = \Omega_n$. We rewrite H as

$$H(\omega) = \sum_{i=1}^{M} \chi_{T_i}(\omega) H(\omega)$$

where $\chi_{T_i}(\omega)$ is the indicator function over region T_i that is 1 when $\omega \in T_i$ otherwise 0. In the Lambertian case, this expands to

$$H(\omega) = \sum_{i=1}^{M} \chi_{T_i}(\omega) L(\omega)(\omega \cdot n) V_p(\omega)$$

If a region T_i is small enough, we can approximate the value of H over T_i as a constant function whose value is the average of H over T_i. Thus we have

$$H(\omega) \approx \sum_{i=1}^{M} \chi_{T_i}(\omega) \frac{\int_{\theta \in T_i} L(\theta)(\theta \cdot n) V_p(\theta) d\theta}{Area(T_i)}$$

where $Area(T_i)$ is the surface area of the region T_i and θ is a dummy variable of integration. If we leave out the visibility term, the integral in the numerator can be interpreted as the response of the environment map defined by $L(\theta)\chi_{T_i}(\theta)$ under the Lambertian BRDF with normal n, which is just the non-shadowed approximation from Equation 2. We can further apply the ambient occlusion approximation as in Equation 3 to approximate the visibility term's contribution with the average occlusion over the region T_i. In other words, let

$$A_i(p) = \frac{\int_{\theta \in T_i} V_p(\theta) d\theta}{Area(T_i)}$$

which gives a value $A_i(p) \in [0,1]$. Since computing $A_i(p)$ exactly is is difficult, we well use an estimate for this value $\hat{A}_i(p)$. Improving the accuracy of the estimate $\hat{A}_i(p)$ will decrease the variance of our final estimator in Equation 7, but to make the estimator unbiased we only require that $\hat{A}_i(p) > 0$ when $A_i(p) > 0$. A piecewise-constant approximation to H, denoted $\hat{H}$, is

$$\hat{H}_{p,n}(\omega) = \sum_{i=1}^{M} \chi_{T_i}(\omega) \frac{\hat{A}_i(p) \int_{\theta \in \Omega} \chi_{T_i}(\theta) L(\theta)(\theta \cdot n) d\theta}{Area(T_i)} \tag{6}$$

$\hat{H}$ will be constant over each region T_i. Like H, $\hat{H}$ varies with the point p and the normal n. Figure 3 shows a plot of $\hat{H}$, incorporating the environment map and the cosine falloff of the Lambertian model.

Our goal is to use the standard importance sampling estimator given in Equation 4 with $\hat{H}_{p,n}$ as the importance function, Ω_n as the domain, and $H_{p,n}$ as the integrand. For this, we need to be able to evaluate $\hat{H}_{p,n}$ efficiently at each pixel, and we need to be able to draw samples from Ω_n according to the distribution $\hat{H}_{p,n}$. We now describe how to accomplish both of these tasks.

The approximation $\hat{H}_{p,n}$ depends on a particular partition of the hemisphere centered around n. Because n may change with every pixel, we must generate a suitable partition on-the-fly, and compute $\hat{H}_{p,n}$ over each of the regions in this partition.

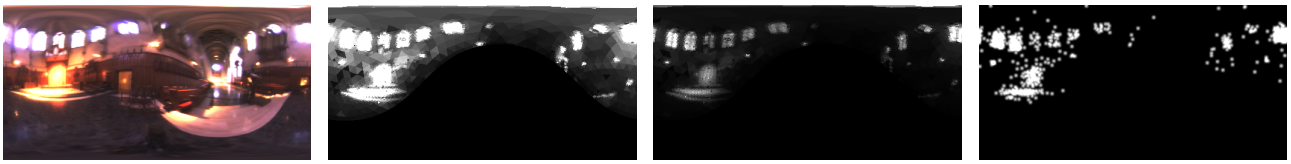
Let $\{T_i\}_{i=1}^M$ be a fixed tessellation of the sphere into M spherical triangles. In Section 4.4 we describe how to generate a good tessellation given an environment map. As a precomputation step that is performed once per environment map, for each spherical triangle T_i, we compute the diffuse convolution of $\chi_{T_i} L$,

$$I_i(n) = \int_{\omega \in T_i} (\chi_{T_i}(\omega) L(\omega)(\omega \cdot n) d\omega$$

and store the results in a texture map indexed by surface normal n. We use a polar coordinate parameterization of the I_i map at 32x64 resolution. (Higher resolutions did not make any difference in rendering quality.) This is the most expensive step in precomputation, and the running time is proportional to M.

In the inner loop of a rendering algorithm, say the renderer needs to estimate reflected radiance at point p with normal n. We generate a tessellation of Ω_n and compute the values of $\hat{H}_{p,n}$ over each spherical triangle in this tessellation as follows. First, we compute a weight W_i for each triangle T_i in the fixed tessellation, $W_i = I_i(n)\hat{A}_i(p)$. $I_i(n)$ is trivial to compute since it was precomputed and stored in a map. Our technique for computing $\hat{A}_i(p)$ will be discussed in Section 4.3.

To generate the tessellation of the hemisphere about n, our idea is to adjust the fixed spherical tessellation by a "clipping" procedure, where each triangle is clipped against Ω_n to generate a *visible triangle list*. The algorithm for spherical triangle clipping is given in Appendix A. The result of clipping a triangle against the hemisphere is that either the entire triangle is outside the hemisphere, in



(a) The "grace" environment map.

(b) A plot of $\hat{H}$ at a surface point.

(c) The same plot as (b) 32 times darker to demonstrate the detail in the bright regions.

(d) The locations of 1000 ray samples chosen according to $\hat{H}$.

Figure 3: Figure (b) and (c) show the value of $\hat{H}$ at a sample point under the grace environment (a). The adaptive tessellation algorithm (Section 4.4) generates higher detail where the environment has higher variance. The ray samples in (d) are concentrated over the brightest light sources, even though they subtend a small solid angle.

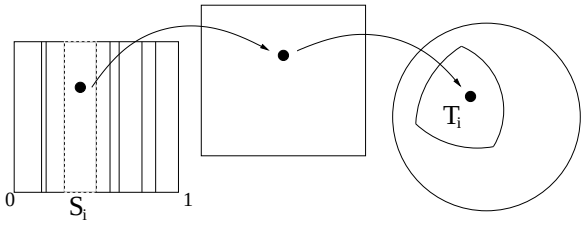


Figure 4: The function m maps from the unit square to the hemisphere according to the weight of each spherical triangle. First, the index of i of a point is determined based on which span S_i its x coordinate lies in. It is mapped to the unit square by normalizing its position within S_i via the *Coord* function, and then to the spherical triangle via the *SphTri* function.

which case it is not included in the visible list, the entire triangle is inside the hemisphere, in which case it is included in the visible list unmodified, or the triangle is partially inside the hemisphere, in which case the triangle must be split.

As described in the Appendix, the portion of a partially clipped triangle T_i that overlaps Ω_n can be expressed as either a smaller triangle T_i', or the union of two smaller non-overlapping triangles T_i^1 and T_i^2. In the case that a single triangle is produced, we include T_i' (but not T_i) in the visible list and set its weight to W_i. In the case that two triangles are produced, we split the weight proportional to the areas[4] of the two result triangles,

$$W_i^1 = \left(\frac{Area(T_i^1)}{Area(T_i)}\right) W_i,\ W_i^2 = W_i - W_i^1.$$

and include T_i^1 and T_i^2 in the visible list (but not T_i). The list of visible triangles is a partition of Ω_n, and the value of $\hat{H}$ over a visible triangle T_i is equal to the triangle's weight W_i divided by its area.

We draw samples from Ω according to $\hat{H}$ as follows. First, for each triangle in the visible list we compute the normalized weight $\overline{W_i}$ as

$$\overline{W_i} = \frac{W_i}{\sum_{T_i visible} W_i}$$

which is the probability that a sample will be drawn from visible triangle T_i. We build a map $m : [0,1] \times [0,1] \to \Omega$ such that if x is uniformly distributed over $[0,1] \times [0,1]$, then $m(x)$ will be distributed over Ω with probability $\hat{H}$. Let M_{vis} be the number of visible triangles. We divide the interval $[0,1]$ into M_{vis} non-overlapping subintervals $S_i = [a_i, b_i)$, such that the length of S_i is $\overline{W_i}$. The function $Idx(x)$ maps from a point in $x \in [0,1]$ to index i such that $x \in S_i$. Idx can be efficiently implemented using a binary search. We also define the map $Coord(i,x)$ as

$$Coord(i,x) = \frac{x - a_i}{b_i - a_i}.$$

The function $SphTri(T,u,v)$, described in [Arvo 1995], uniformly maps from the unit square to the spherical triangle T. Finally, m is defined as

$$m(u,v) = SphTri(T_{Idx(u)}, Coord(Idx(u),u), v).$$

Figure 4 gives a schematic view of m. When $Idx(u) = i$, $m(u,v) \in T_i$, and the value of $\hat{H}(m(u,v))$ is $W_i/Area(T_i)$.

If $\{x_j\}_{j=1}^N$ are random samples uniformly distributed over $[0,1] \times [0,1]$, the environment-based estimator for Equation 1 is[5]

$$\hat{I}_{\hat{H},N}(p,n) = \frac{1}{N} \sum_{j=1}^{N} \frac{L(m(x_j))(m(x_j) \cdot n) V_p(m(x_j))}{\hat{H}(m(x_j))} \qquad (7)$$

While m has the desired sampling property, it is discontinuous and hence will destroy any stratification structure of the point set $\{x_j\}$. Despite this, we have observed that points $\{x_j\}$ generated using randomized QMC sequences give better results than independently drawn samples.

4.3 The shadow cache

The visibility term makes Equation 1 very difficult to solve because it makes the integrand discontinuous. However, as shown in Figure 5, there is coherence in the values of V_p within a neighborhood of p.

Our method for estimating $A_i(p)$ takes advantage of this coherence. Whenever a hemispherical ray is cast, we note from which triangle T_i in the fixed spherical tessellation the sample was taken, and record whether the ray was occluded or not. For each triangle T_i, we compute a denominator d_i equal to the total number of rays

[4] Girard's Theorem says that the area of a spherical triangle is $\alpha + \beta + \gamma - \pi$ where α, β and γ are the dihedral angles at the triangle's vertices.

[5] We are glossing over the issue of multiple spectral wavelengths. Ideally, we would sample each wavelength separately. In practice this is too expensive, so we use use the luminance value of $I_i(n)$ for the calculation of W_i.

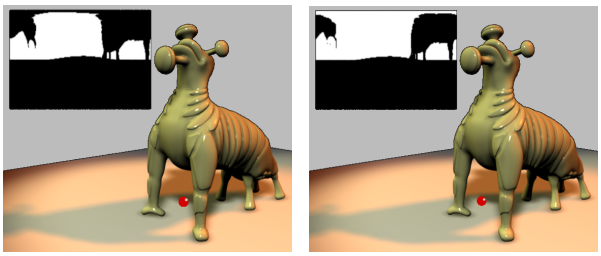


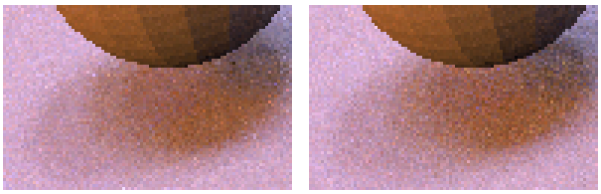
Figure 5: The shadow cache exploits the principle that the visibility function (plotted in the upper left corner with a polar parameterization) is similar for points that are near each other.

cast through that triangle, and a numerator n_i equal to the number of those rays that were unoccluded.[6] To save memory, the shadow cache values can be stored at a lower angular resolution than the fixed tessellation used in $\hat{H}$, as long as there is a correspondence between the indices. We call a complete set of values n_i and d_i centered about a particular point a *cache item*. Cache items are stored in a data structure called the *shadow cache*. To compute $\hat{A}_i(p)$, we search the shadow cache for all cache item nearby to p, and compute the sums of the items' numerators and denominators, σn_i and σd_i. If σd_i is less than some threshold (we use 3), we say that there is not enough information to make a good estimate and use a value of 1. Otherwise, we use $\sigma n_i / \sigma d_i$.

Rather than using this value for $\hat{A}_i(p)$ directly in Equation 6, we remap the value following defensive importance sampling [Owen and Zhou 2000; Hesterberg 1995] according to a tunable parameter $\alpha \in [0,1]$,

$$\alpha \hat{A}_i(p) + (1-\alpha)(.5)$$

which prevents $\hat{H}$ from having an arbitrarily small tail distribution. We have gotten good results with $\alpha = .7$.



(a) With shadow cache. (b) No shadow cache.

Figure 6: (a) was rendered with a shadow cache, (b) was rendered without. The error in the shadow region is reduced by 10 percent.

We have implemented a simple image-space caching data structure, which stores a cache item every few pixels, similar to the irradiance cache [Ward et al. 1988]. Proximity queries are then based on image-space locality. This image space locality condition may result in noisier regions near geometric discontinuities as shadow cache information is propagated from one pixel to a neighbor even though the corresponding points on the surface of the scene are not spatially close. We believe that these artifacts are not a fundamental flaw of the shadow cache, however, and could be alleviated by testing for spatial proximity before sharing cache items between

[6] Since we only need approximate shadow information, n_i and d_i are stored with 8 bits precision each.

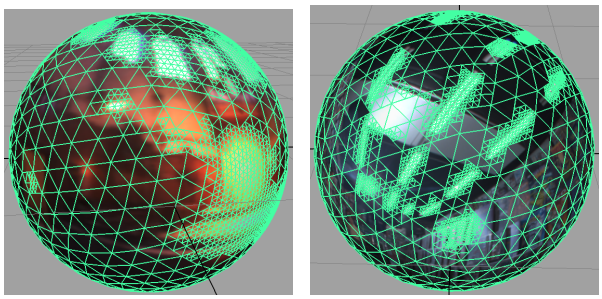


Figure 7: Triangulations with $M = 5000$ generated from grace environment (a) and the video store interior (b). The minimum subdivision level is 3, the maximum is 6.

pixels, or by storing the shadow cache in a more sophisticated spatial data structure such as a kd-tree or mapped onto the surface of the geometry. With our implementation, the shadow cache typically reduces error by about 10 percent in shadowed regions, as shown in Figure 6.

The value of $A_i(p)$ is a purely geometric quantity, and hence could be baked into the scene geometry as a preprocess or reused between renders if the geometry does not change.

While it may seem that the shadow cache results in a biased algorithm, this is not the case. In fact, the technique presented here is truly unbiased. The proof is as follows.

The standard importance sampling estimator is unbiased if two conditions on the importance function are met: (1) the importance function is non-zero whenever the integrand is non-zero, and (2) the samples are all drawn with respect to the same importance function. Condition (1) is met because a shadow cache values of 0 is never used as described above. Condition (2) is met by design. In the initialization stage before evaluating the integral for a given pixel, the algorithm reads the values in the shadow cache for the current pixel and uses these values to build the importance function $\hat{H}$. In the course of evaluating the integral, the renderer cast rays into the scene to test occlusion. The results of these rays casts are stored in the shadow cache. However, the adjustments made to the shadow cache do not feed back into the *current* integral evaluation, but are only used in *subsequent* evaluations of reflected radiance at nearby pixels. Therefore each individual evaluation of reflected radiance per pixel is unbiased. The shadow cache merely accelerated convergence of the Monte Carlo estimator, but does not otherwise affect the result.

4.4 Choosing a good triangulation

As stated in Equation 5, the variance of an estimator is the integral of the ratio of the squared residual to the importance function. Thus there will be high variance when the residual is high relative to the estimated importance. In general, this means the variance of our estimator will be high when $L(\omega)(\omega \cdot n)V_p(\omega)$ varies a lot over the area of a particular triangle. Because $\omega \cdot n$ and $V_p(\omega)$ depend on n and p, there is no way to take them into account when devising the fixed tessellation of the sphere. Instead, the best we can do reduce the residual error is to minimize the variation of $L(\omega)$ over each triangle by more finely tessellating regions of the sphere where $L(\omega)$ is varying the most.

We use a greedy algorithm to generate adaptive triangulations of the sphere. We begin with an icosahedron, which has 20 triangles, and subdivide a minimum number of times (a minimum level of three works well). All triangles are placed in a heap sorted in decreasing order by the variance of the environment map $L(\omega)$ over

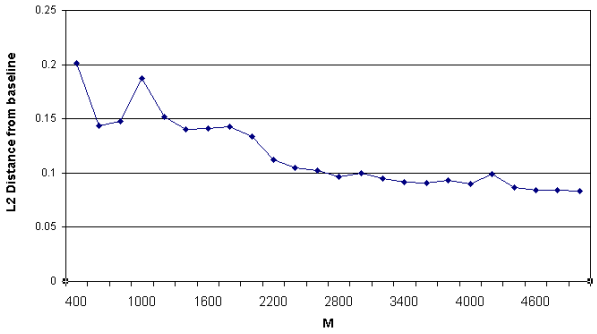


Figure 8: Plot of L^2 error of rendering a scene under the grace environment as a function of M.

the region of that triangle. Given a triangle budget M, we iteratively remove the first triangle from the heap (which will have the largest variance) and test if its subdivision level is below a user threshold. If so, we subdivide it once, and place the four new child triangles back on the heap. The process halts when there are M total triangles. The results of this algorithm for the grace and video store environments are shown in Figure 7.

As future work, we would like to derive optimal upper and lower bounds on the tesselation level of the icosahedron based on the surface BRDF and the variance in the environment map.

M is a quality setting, and we can afford to set it quite high without adversely affecting rendering times. The per-hemispherical sample cost if $O(log(M))$, which is the cost of the Idx function. The per-pixel cost if $O(M)$, but as the number of ray samples increases, this cost is quickly overtaken by the cost of ray casting. We performed an experiment where we rendered the scene in Figure 1 at 10,000 samples per pixel as our baseline. We then rendered the scene with different values of M at 165 samples per pixel, and computed the L^2 distance between the rendered image and the baseline image. The plot of this error is shown in Figure 8. The error decreases significantly up until about 2200, then levels off around $M = 3500$, although it continues decreasing slightly. For all the renders in the paper and in the video, we set M at 5000.

5 Results and Comparisons

The images in Figure 9 compare the new sampling scheme against cosine-weighted QMC integration. The video environment (Figure 9(a)) was captured on a live-action set. The outdoor environment (Figure 9(b)) was captured in direct sunlight which is difficult to sample because the sun dominates, yet subtends a small solid angle. The office environment (Figure 9(c)) was captured in an office and has no natural light.

We rendered the armadillo model in these three environments at a sampling rate so that a small amount of noise is still visible. The video store environment is the hardest to sample because there are two very bright and small lights that dominate the scene. Our sampling scheme effectively reveals hard shadows from the key lights and soft shadows from the fluorescents (Figure 9(d)), while cosine-weighted sampling fails completely (Figure 9(g)). The results from the outdoor environment are similar (Figures 9(e) and (f)).

We rendered the video and outdoor scenes at 4000 samples per pixel with cosine-weighted sampling, and the renders were still extremely noisy. In these cases, our sampling scheme is at least 15 and 20 times more efficient, respectively. For the office environment, cosine-weighted sampling matches the noise level of Figure 9(f) at 1200 samples per pixel, indicating our scheme is 10 times more efficient in this case.

In general, the greater the dynamic range in the environment map, the more environment-based importance sampling out performs cosine-weighted sampling, in some cases by well over an order of magnitude.

6 Alternate Techniques

In the course of developing this algorithm, we compared it against several other possibilities.

6.1 Defensive Importance Sampling

We originally implemented a more sophisticated technique, Hesterberg's defensive importance sampling [Hesterberg 1995]. As the importance function, we used a mixture between the environment-based importance ($\hat{H}$) and a uniform distribution. (For notation's sake, assume $\hat{H}$ has been normalized to have unit integral over the hemisphere.) In other words, take the importance function to be

$$Imp(\omega) = \beta\hat{H}(\omega) + (1-\beta)\frac{1}{2\pi}$$

where β is the "defensive parameter" that mixes in a distribution with a broad tail, in this case the uniform distribution. A non-zero value of β will prevent samples from clustering in bright areas only. Interestingly, best results were obtained with $\beta = 0$, *i.e.*, no defensive sampling. Figure 10 shows graphs of the L^2 error of the floating sphere rendered under the grace environment as function of β.

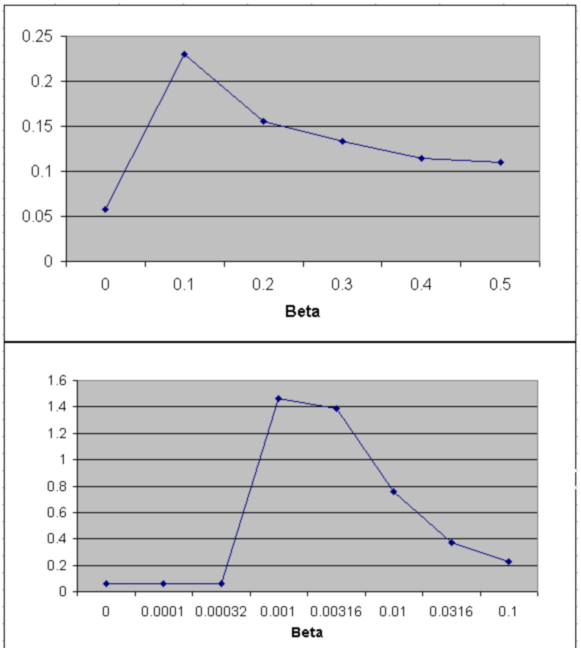


Figure 10: Total image L^2 error under the "grace" environment as a function of β. The top graph is a linear graph in the range $[0, 0.5]$, the bottom is log-linear in the range $[0, 0.1]$.

Our guess is that pure importance sampling works best because occlusion of non-important lightsources is not a major source of noise. Even if the importance function does not match the integrand in a dark region due to occlusion, the importance function will *overestimate* the integrand, which is not nearly as bad as *underestimating* the integrand. As long as the bright regions are sampled well, the estimator will still have low variance.

(a) Video store environment (b) Outside environment (c) Office environment

(d) Environment-based sampling, 256 samples. (e) Environment-based sampling, 200 samples. (f) Environment-based sampling, 128 samples.

(g) Cosine-weighted QMC, 256 samples. (h) Cosine-weighted QMC, 200 samples. (i) Cosine-weighted QMC, 128 samples.

Figure 9: Comparison of the new sampling technique against QMC cosine weighted sampling under different lighting environments.

We believe that mixed importance sampling/control variates schemes along the lines of [Owen and Zhou 2000] may work better than pure importance sampling, but investigating whether this is so remains future work.

6.2 Pixel-based Importance Sampling

We implemented a standard importance sampling estimator where the importance function is derived from a lower resolution luminance version of the actual envionment map. Call this low-res image $E[.,.]$ with dimension $w \times h$, and assume it is a standard polar parameterization, with the positive Y-direction corresponding to the top row in E. Say we are shading a surface point with normal n with Lambertian reflectance. Given a pixel (i, j) corresponding to direction ω in the environment map, the probability of drawing a sample from that pixel should be proportional to

$$Prob(i, j) \sim H(i, j) = max(\omega \cdot n, 0)E[i, j]Area(i, j)$$

where $Area(i, j)$ is the area subtended by the pixel, which is approximated by $4\pi^2\sqrt{1-\omega_y}/wh$.

For each pixel in E, compute $H(i, j)$, and take the sum over all pixels, $S = \sum_{i,j} H(i, j)$. The probability of choosing pixel (i, j) is equal to $H(i, j)/S$, and the value of the importance p.d.f over pixel (i, j) is $2\pi H(i, j)/(Area(i, j) \cdot S)$. We use a chart similar to m to distribute samples over the hemisphere according to this importance function. From these building blocks, it is straightforward to implement standard importance sampling to estimate the reflected radiance. This method is $O(wh)$ per pixel, and $O(log(wh))$ per ray sample.

The triangular-based approach in Section 4 significantly outperforms this pixel-based approach in terms of efficiency and bias. First of all, the pixel-based method is biased. This is because we are approximating the hemisphere centered around n with the union of pixel regions, which will not match a hemisphere exactly. As a result, ray samples will sometimes be generated in a direction that is in the list of visible pixels, yet outside the visible hemisphere. To

generate an unbiased estimator, we would need to "clip" the pixel's polygonal support to the hemisphere, similar to the triangle-based scheme. However, this is much more elegant to handle in the case of triangles since clipping triangles yields more triangles, while clipping 4-sided polygons may yield more complicated shapes.

In terms of efficiency, it is better to use an adaptive tesselation of the sphere than a fixed grid such as in a pixel-based scheme. The running time of both algorithms depend on the number of subdivisions of the sphere, M in the case of triangles, or $w \cdot h$ in the case of pixels. Because it is more efficient to use an importance function that matches the integrand with fewer subdivisions of the sphere, the adaptive scheme outlined in the paper is more efficient than pixel-sampling. One could use a multiresolution representation of the environment map, but the spherical triangle scheme handles multiresolution with no extra complexity.

Figure 11 shows 4 images. All were rendered with 165 samples per pixel. Rendering times were all comparable (within 20 percent). The images visually demonstrate that an adaptive triangular subdivision of the sphere yields superior results in the same rendering time. Numerical results are listed in Table 6.2. The L^2 error in the table is computed as the image distance from a "baseline" render at a very high sampling rate. The data shows that in the case of the complex "video store" environment, the scheme in the paper is much more efficient than a naive pixel-based approach. The two schemes are more evenly matched for the grace environment, although the triangle scheme is still better. The rendering time measurements are taken from a non-optimized prototype renderer and should be taken as suggestive only.

7 Future Work

There are a number of areas where the sampling scheme could be improved. As stated above, the function m does not preserve stratification structure of an input point set. A mapping that does a better job of this could better leverage advantages of QMC integration, which could further decrease the variance of the estimator. Higher-order approximations to H might give better results by matching the integrand H more accurately. The difficulty with higher-order functions on the sphere is that it can be quite difficult use them as a p.d.f. from which to draw samples. Using the approach outlined in [Arvo 2001], it is very difficult to derive a closed-form expression for a map from the unit square to a spherical triangle according to a non-constant p.d.f. A scheme such as rejection sampling might work, but that is generally not compatible with stratified sampling.

We believe there is potential for greatly improving the efficiency of this technique by more clever exploitation of the shadow cache. Here, we present only a simple version. More sophisticated and accurate shadow caching techniques, and more accurate ways of extracting information from them, could potentially yield a significantly more efficient estimator when occlusion is complex, such as for a finely displacement-mapped character.

Also, in a probabilistic framework such as presented here, it is easy to take advantage of approximate information about where light is distributed to increase efficiency. For example, we could incorporate information from a Photon map (as in [Jensen 1995]), in order to trace rays where we expect to find either direct or indirect light. Also, it would be straightforward to use a more complex visibility function that accounts for transmission through translucent media.

References

ARVO, J. 1995. Stratified sampling of spherical triangles. In *Proceedings of SIGGRAPH 95*.

ARVO, J. 2001. Stratified sampling of 2-manifolds. In *Course Notes for State of the Art in Monte Carlo Ray Tracing for Realistic Image Synthesis*, SIGGRAPH 2001.

BLINN, J. F., AND NEWELL, M. E. 1976. Texture and reflection in computer generated images. *Communications of the ACM 19*, 10 (Oct.), 542–547.

CHRISTENSEN, P. H. 2002. Note 35: Ambient occlusion, image-based illumination, and global illumination. *PhotoRealistic RenderMan Application Notes*.

COHEN, J. M., AND DEBEVEC, P. E., 2001. "LightGen" HDRShop plugin. http://www.ict.usc.edu/∼jcohen/lightgen/lightgen.html.

COOK, R. L., PORTER, T., AND CARPENTER, L. 1984. Distributed ray tracing. In *Proceedings of SIGGRAPH 84*, 137–145.

DEBEVEC, P. E., AND MALIK, J. 1997. Recovering high dynamic range radiance maps from photographs. In *Proceedings of SIGGRAPH 97*.

DEBEVEC, P. E. 1998. Rendering synthetic objects into real scenes: Bridging traditional and image-based graphics with global illumination and high dynamic range photography. In *Proceedings of SIGGRAPH 98*.

DUTRE, P., 2001. Global illumination compendium: The concise guide to global illumination algorithms. http://www.cs.kuleuven.ac.be/∼phil/GI/.

EVANS, M., AND SWARTZ, T. 2000. *Approximating Integrals via Monte Carlo and Deterministic Methods*. Oxford University Press.

GREENE, N. 1986. Environment mapping and other applications of world projections. *IEEE Computer Graphics and Applications 6*, 11 (Nov.).

HESTERBERG, T. 1995. Weighted averge importance sampling and defensive mixture distributions. *Technometrics 37*, 2, 185–194.

JENSEN, H. W., AND CHRISTENSEN, N. J. 1995. Efficiently rendering shadows using the photon map. In *Compugraphics '95*, 285–295.

JENSEN, H. W. 1995. Importance driven path tracing using the photon map. In *Proceedings of Eurographics Rendering Workshop 95*, 326–335.

JENSEN, H. W. 2001. *Realistic Image Synthesis Using Photon Mapping*. AK Peters.

KATO, T. 2002. Photon mapping at SquareUSA: The kilauea renderer. In *Course Notes for A practical guide to global illumination using photon mapping*, SIGGRAPH 2002.

KAUTZ, J., AND MCCOOL, M. D. 1999. Interactive Rendering with Arbitrary BRDFs using Separable Approximations. In *Tenth Eurographics Rendering Workshop 1999*, Eurographics, D. Lischinski and G. W. Larson, Eds., 281–292.

KELLER, A. 2002. Hierarchical monte carlo image synthesis. *Mathematics and Computers in Simulation 55*, 1–3, 79–92.

KOLLIG, T., AND KELLER, A. 2002. Efficient illumination by high dynamic range images. Tech. Rep. 323/02, University of Kraiserslauten.

KOLLIG, T., AND KELLER, A. 2002. Efficient multidimensional sampling. In *EUROGRAPHICS 2002*, vol. 21.

LAFORTUNE, E. P., AND WILLEMS, Y. D. 1993. Bi-directional path tracing. In *Proceedings of Compugraphics '93*, 145–153.

LATTA, L., AND KOLB, A. 2002. Homomorphic factorization of brdf-based lighting computation. In *Proceedings of SIGGRAPH 2002*.

MILLER, G. S., AND HOFFMAN, C. R. 1984. Illumination and reflection maps: Simulated objects in simulated and real environments. In *Course Notes for Advanced Computer Graphics Animations*, SIGGRAPH 1984.

OWEN, A., AND ZHOU, Y. 2000. Safe and effective importance sampling. *Journal of the American Statistical Association 95*, 450 (June).

RAMAMOORTHI, R., AND HANRAHAN, P. 2001. A signal-processing framework for inverse rendering. In *Proceedings of SIGGRAPH 2001*, 117–128.

SHIRLEY, P., WANG, C., AND ZIMMERMAN, K. 1996. Monte carlo methods for direct lighting calculations. *ACM Transactions on Graphics* (Jan.).

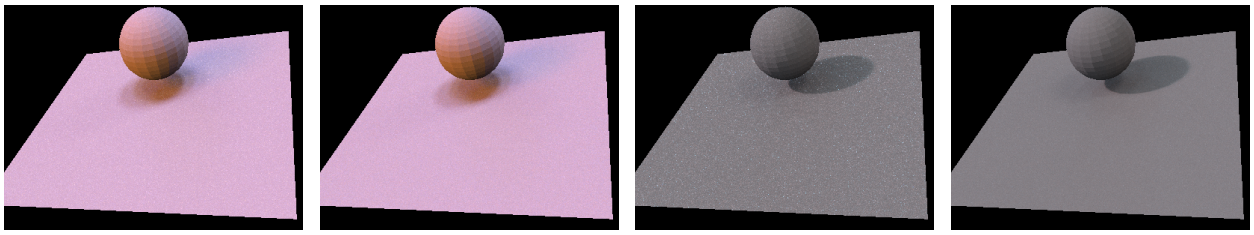
VEACH, E., AND GUIBAS, L. J. 1995. Optimally combining sampling techniques for monte carlo rendering. In *Proceedins of SIGGRAPH 1995*, 419–428.

VEACH, E., AND GUIBAS, L. J. 1997. Metropolis light transport. In *Proceedings of SIGGRAPH 1997*, 65–76.

VEACH, E. 1997. *Robust Monte Carlo Methods for Light Transport Simulation*. PhD thesis, Stanford.

WARD, G. J., RUBINSTEIN, F. M., AND CLEAR, R. D. 1988. A ray tracing solution to diffuse interreflection. In *Proceedings of SIGGRAPH 88*, 85–92.

WILLIAMS, L. 1983. Pyramidal parametrics. In *Proceedings of SIGGRAPH 1983*, 1–11.



(a) "Grace" environment, pixel-based subdivision with $w = 200$, $h = 100$.

(b) "Grace" environment, triangle-based subdivision with 5000 triangles.

(c) "Video store" environment, pixel-based subdivision with $w = 200$, $h = 100$.

(d) "Video store" environment, triangle-based subdivision with 5000 triangles.

Figure 11: Comparison of pixel-based sampling versus triangle-based sampling.

Env Map Resolution	Spherical subdivisions	Render time	L^2 Error, "grace"	L^2 Error, "video store"
100 x 50	5,000	177 sec	0.96255	1.744533
150 x 75	11,250	283 sec	0.80915	1.462526
180 x 90	16,200	379 sec	0.78055	1.290449
200 x 100	20,000	449 sec	0.62211	1.135643
5000 Triangles	5,000	390 sec	0.58214	0.367410

Table 1: Comparison of running time to image error under the grace and video store environments, using a naive pixel-based scheme. The bottom row is the triangle-based scheme outlined in the paper. All images were rendered at 165 samples per pixel.

A Clipping Spherical Triangles

Given a spherical triangle ABC defined by vertices A, B, and C, and a hemisphere centered about n, Ω_n, we want to compute whether the triangle overlaps the hemisphere, and if so, to describe the region of overlap. First, we test if the dot product of each vertex with n is positive or negative. Let t be the number of vertices with positive dot product.

If $t = 0$, the triangle is fully outside the hemisphere.

If $t = 3$, the triangle is fully inside the hemisphere.

If $t = 2$, then the triangle partially overlaps and the intersection of ABC with Ω_n can be described as the union of two spherical triangles. Let A be the vertex that such that $A \cdot n < 0$. Let D be the point where the great arc that bounds Ω_n intersects the great arc segment AB, $D = \partial\Omega_n \cap AB$. This is computed as $D = \pm n \times (A \times B)$, where the sign is chosen so that D lies between A and B. Similarly let $E = \partial\Omega_n \cap AC$. Then the two resulting triangles are BCD and CED.

If $t = 1$, then the triangle partially overlaps Ω_n and the intersection of ABC with Ω_n is a single spherical triangle. Let A be the vertex such that $A \cdot n > 0$. Define D and E as above. The resulting triangle is ADE.