Population Monte Carlo Samplers for Rendering

ShaoHua Fan*
University of Wisconsin-Madison

Yu-Chi Lai†
University of Wisconsin-Madison

Stephen Chenney‡
Emergent Game Technology

Charles Dyer§
University of Wisconsin-Madison

Figure 1: A room scene with complex models computed using photon mapping with both an adaptive image plane sampler and an adaptive hemispheric integral sampler.

Abstract

We present novel samplers and algorithms for Monte Carlo rendering. The adaptive image-plane sampler selects pixels for refinement according to a perceptually-weighted variance criteria. The hemispheric integrals sampler learns an importance sampling function for computing common rendering integrals. Both algorithms, which are unbiased, are derived in the generic Population Monte Carlo statistical framework, which works on a population of samples that is iterated through distributions that are modified over time. Information found in one iteration can be used to guide subsequent iterations. Our results improve rendering efficiency by a factor of between 2 to 5 over existing techniques. We also show how both samplers can be easily incorporated into a global rendering system.

CR Categories: I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism—Color, shading, shadowing, and texture

Keywords: Population Monte Carlo, adaptive rendering, mixture sampling, direct lighting, adaptive sampling

1 Introduction

Monte Carlo integration methods offer the most general solution to physically accurate lighting simulation. For production applications, algorithm efficiency is of primary concern: image noise (variance) must be low at practical computation times. We present sampling techniques that significantly improve rendering efficiency for image-plane sampling and hemispheric integrals. Both are derived using the Population Monte Carlo (PMC) sampling framework, which is a technique that adapts sampling distributions over time and enables sample re-use, all with theoretical guarantees on error and little computational overhead.

PMC algorithms iterate on a population of samples. In our simplest sampler, for image-plane sampling (PMC-IP), the population is a set of image-plane locations. The population is initialized in some way, say using stratified sampling, then PMC-IP generates an initial image. Any information available at this stage can then be used to adapt a *kernel function* that produces a new population. In image-plane sampling, the perceptually-weighted variance in the intermediate images is used to construct the kernel function, resulting in more image plane samples in regions of high variance. The procedure is then iterated: sample, adapt, sample,.... The result is an unbiased adaptive algorithm.

*e-mail: shaohua@cs.wisc.edu
†e-mail: yu-chi@cs.wisc.edu
‡e-mail: schenney@gmail.com
§e-mail: dyer@cs.wisc.edu

In the case of direct lighting, or hemispheric integrals in general, importance sampling [Pharr and Humphreys 2004] is the primary variance reduction tool. However, a poor choice of importance function can *increase* variance, and, moreover, the best importance function can vary throughout a rendering depending on such things as surface properties, lighting configurations and the presence of shadows. For example, the ideal importance function for a semi-gloss surface depends on whether the primary lobe points toward a light source, or the surface is in shadow, or neither. These configurations vary over a surface and they are difficult to discover before sampling begins, yet the choice of importance functions is typically made once and remains fixed. PMC for hemispheric integrals (PMC-HI) improves sampling efficiency by dynamically choosing importance functions based on information gathered during rendering.

Photon mapping and path tracing have been the industrial rendering algorithms for global illumination. Our two sampling methods, PMC-IP and PMC-HI can improve the efficiency of these two algorithms with minimal modifications to the rendering system. Furthermore, a complete rendering system enables further improvements to PMC-HI.

Population Monte Carlo is a general purpose framework with many variants. The challenge in applying it to rendering lies in the small sample counts, the hard-to-evaluate distributions, and the visual sensitivity to noise. Our contribution is two specific tools for rendering that use the framework:

- An **Image-Plane Sampler**, PMC-IP, that adapts to guide samples to perceptually high variance image regions, is cheap to compute, maintains stratification, and is unbiased.
- An **Hemispheric Integral Sampler**, PMC-HI, that adjusts the sampling directions used to evaluate hemispheric integrals at a point and supports a variety of importance functions acting together. We can, for instance, avoid over-sampling a light source from a surface point within its shadow, or a BRDF specular lobe that makes no contribution. Furthermore, we can guide samples toward important illumination directions found by previous samples, without adding bias.

We include results comparing each algorithm to existing approaches. We find that PMC-based algorithms improve the efficiency in a factor of 2 to 5 over existing methods. We also show how to incorporate our algorithms into a modern rendering system. The algorithms are independent of each other. However, in a complete rendering system information can be fed from each iteration of the image plane sampler back to the PMC-HI sampler to gain even greater improvement in efficiency.

2 Related Work

Here we focus on two specific areas of related work: adaptive image-plane sampling, and sampling for hemispheric integrals. For an overview of Monte Carlo rendering in general, see Pharr and Humphreys [2004].

Typically, adaptive image-plane algorithms perform a first pass with a small number of samples per pixel and use the resulting values to label pixels as adequately sampled or in need of further refinement. The algorithm then iterates on the pixels requiring more samples [Glassner 1995; Painter and Sloan 1989; Purgathofer 1986; Mitchell 1987; Bolin and Meyer 1998; Ramasubramanian et al. 1999].

A common property of these existing algorithms is that they stop sampling a given pixel when some image-derived metric is satisfied. As Kirk and Arvo [Kirk and Arvo 1991] point out, the evaluation of the image metric relies on random samples, so there is some non-zero probability that the threshold is incorrectly detected and that sampling stops too soon. This introduces bias in the final image, which is a problem when physically accurate renderings are required. Our algorithm never uses a threshold to stop sampling a pixel, and it is statistically unbiased.

Many metrics have been proposed for the test to trigger additional sampling. Lee et al. [1985] used a sample variance based metric. Dippé and Wold [1985] estimated the change in error as sample counts increase. Painter and Sloan [1989] and Purgathofer [1986] used a confidence interval test, which Tamstorf and Jensen [1997] extended to account for the tone operator. Mitchell [1987] proposed a contrast based criteria because humans are more sensitive to contrast than to absolute brightness, and Schlick [1991] included stratification into an algorithm that used contrast as its metric. Bolin and Meyer [1998], Ramasubramanian et al. [1999] and Farrugia and Péroche [2004] used models for human visual perception, of which we use a variant. Most recently, Rigau et al. [2002; 2003] introduced entropy-based metrics.

Our algorithm views the image plane as a single sample space for the purposes of sampling. Dayal et al. [2005] took a similar view in the context of frameless rendering. They used a variance-based metric to control a kD-tree subdivision where samples are drawn uniformly within each adaptively sized cell of the subdivision. Stokes et al. [2004] also took a global approach with their perceptual metric.

There is a large body of work on computing hemispheric integrals (direct lighting), mostly concerned with importance sampling functions. Veach's thesis [1997] provides a description of the basic methods and analysis of variance. Importance functions are commonly based on surface BRDFs (see Pharr and Humphreys [2004] for an overview of these), or light sources [Shirley et al. 1996; Agarwal et al. 2003]. Recent advances include wavelet-based importance functions for environmental lighting [Clarberg et al. 2005], and resampling algorithms [Burke et al. 2005; Talbot et al. 2005] that avoid visibility queries for samples that are likely to be unimportant. However, the former is applicable only to environment maps, while the latter throws away samples and still requires a-priori choice of importance functions. No existing importance sampling approach for hemispheric integrals offers adaptable importance functions.

Work on adaptive PDFs for importance sampling has focused on path tracing or irradiance caching applications. Dutré and Willems [1994] used piecewise linear functions to determine shooting directions out of light sources in a particle tracing application. Dutré and Willems [1995] use piecewise constant functions and Pietrek and Peter [1999] use wavelets to build adaptive PDFs for sampling gather directions in path tracing. A diffuse surface and piecewise constant PDF assumption is required to reduce the number of coefficients to a manageable level, and even then very high sample counts are required. It is important to note that a bad approximation can *increase* variance. Lafortune and Willems [1995] used a 5D tree to build an approximation to radiance in the scene, and then use it for importance sampling in a path tracing framework. The same problems with sample counts and approximation errors arise in their work. Our algorithm works with arbitrary BRDFs and uses a low-parameter adaptive model to minimize the sample count required to control adaption.

Adaptive algorithms have also been suggested for shadow computations. Ward [1991] proposed an algorithm for scenes with many

lights, where shadow tests for insignificant lights are replaced by probabilistic estimates. Ward's approach works best with many light sources (tens or hundreds) while our technique works best with few sources. Ohbuchi and Aono [Ohbuchi and Aono 1996] adaptively sampled an area light source (which introduces bias). They achieved good stratification by employing quasi-Monte Carlo (QMC) techniques to place the samples, a technique we also use.

A Sequential Monte Carlo algorithm, similar in spirit to Population Monte Carlo, has recently been applied by Ghosh, Doucet and Heidrich [2006] to the problem of sampling environment maps in animated sequences. Their work exploits another property of iterated importance sampling algorithms – the ability to re-use samples from one iteration to the next – and is complementary to our approach.

3 Population Monte Carlo (PMC)

The Population Monte Carlo algorithm [Cappé et al. 2004] is an iterated importance sampling scheme. In this scheme, a sample population approximately distributed according to a target distribution is generated at each iteration. Then the samples from all the iterations can be used to form unbiased estimates of integrals under that distribution. It is an adaptive algorithm that calibrates the proposal distribution to the target distribution at each iteration by learning from the performance of the previous proposal distributions.

Assume we have a population of samples denoted by $\left\{X_1^{(t)},\ldots,X_N^{(t)}\right\}$, where t is the iteration number and N is the population size, and we wish to sample according to the distribution proportional to $f(x)$. The generic PMC sampling algorithm is stated in Figure 2.

1 generate the initial population, $t=0$
2 **for** $t=1,\cdots,T$
3 adapt $K^{(t)}(x^{(t)}|x^{(t-1)})$
4 **for** $i=1,\cdots,N$
5 generate $\hat{X}_i^{(t)} \sim K^{(t)}(x|X_i^{(t-1)})$
6 $w_i^{(t)} = f(\hat{X}_i^{(t)})/K^{(t)}(\hat{X}_i^{(t)}|X_i^{(t-1)})$
7 resample according to $w_i^{(t)}$ for the new population

Figure 2: The generic Population Monte Carlo algorihtm.

Line 1 creates the population to jump-start the algorithm. Any method can be used to generate these samples provided that any sample with non-zero probability under f can be generated, and the probability of doing so is known.

The outer loop is over iterations. In each iteration of the algorithm, a *kernel function*, $K^{(t)}(x^{(t)}|x^{(t-1)})$, is determined (line 3) using information from the previous iterations. The kernel function is responsible for generating the new population, given the current one. It takes an existing sample, $X_i^{(t-1)}$, as input and produces a candidate new sample, $\hat{X}_i^{(t)}$, as output (line 5). The distinguishing feature of PMC is that the kernel functions are modified after each step based on information gathered from prior iterations. The kernels adapt to approximate the ideal importance function based on the samples seen so far. While this dependent sampling may appear to introduce bias, it can be proven that the result is either unbiased or consistent, depending on whether certain normalizing constants are known (in our case they are known).

The weight computed for each sample, $w_i^{(t)}$, is essentially its importance weight. The resampling step in line 7 is designed to cull candidate samples with low weights and promote high-weight samples. Resampling is not always necessary, particularly if the kernel is not really a conditional distribution. In our two sampling algorithms, we did not use the resampling step.

At any given iteration, an estimator of the integral of interest is

$$\int_{\mathscr{D}} f(x)dx = \frac{1}{N}\sum_{i=1}^{N} w_i^{(t)} \tag{1}$$

As with importance sampling, this estimator uses the sample weights to estimate the normalization constant, Z, of the the target distribution $\pi = \frac{1}{Z}f(x)$. In practice, we can average over all iterations to improve the estimate, even weighting each iteration differently if we choose. The detail in proving the unbiasedness and analyzing the variance of PMC method is given in [Anonymous 2006].

Several steps are required to apply PMC to rendering problems:

- Decide on the sampling domain and population size. Computational concerns and stratification typically drive the choice of domain. In the image-plane case, working on a discrete pixel domain rather than a continuous one makes stratification simpler to implement and sampling more efficient. We discuss the choice of population size in the context of each algorithm, and later in the discussion.
- Define kernel functions and their adaption criteria. This is the most important task, and we give examples for our applications and suggest some general principles in the discussion. For rendering applications two key concerns are the degree to which the kernel supports stratification and whether it works with a small population size (as low as 4 in our hemispheric integrals sampler).
- Choose the techniques for sampling from the kernel functions and the resampling step. The deterministic sampling we use significantly reduces variance much like stratification.

The following sections describe each of our samplers in detail, adaptation of them into a modern rendering system, and a general discussion on PMC for rendering problems before we conclude with results.

4 PMC-IP: Image-Plane Sampling

Physically-based rendering algorithms compute the intensity, $I(i,j)$, of each pixel (i,j), by estimating the integrals:

$$I_{i,j} = \int_{\mathscr{I}} W_{i,j}(\mathbf{u})L(\mathbf{x},\omega)d\mathbf{u} \tag{2}$$

where $\mathscr{I}$ is the image plane, $W_{i,j}(\mathbf{u})$ is the measurement function for pixel (i,j) – non-zero if $\mathbf{u}$ is within the support of the reconstruction filter at (i,j) – and $L(\mathbf{x},\omega)$ is the radiance leaving the point, $\mathbf{x}$, seen through $\mathbf{u}$ in the direction $-\omega$, determined by the projection function of the camera. We are ignoring, for discussion purposes, depth of field effects, which would necessitate integration over directions out of the pixel, and motion blur, which would require integration over time.

An image-plane sampler selects the image-plane locations, $\mathbf{x}$ in Equation 2. For simplicity, assume we are working with a ray-tracing style algorithm that shoots from the eye out into the scene.

Adaptive sampling aims to send more rays through image locations that have high noise, while avoiding bias in the final result.

Taking an importance sampling view, given a set of samples, $\{\mathbf{X}_1,\ldots,\mathbf{X}_N\}$ from an importance function $p(\mathbf{x})$, each pixel is estimated using

$$\hat{I}_{i,j} = \frac{1}{n}\sum_{k=1}^{N}\frac{W_{i,j}(\mathbf{X}_k)L(\mathbf{X}_k,\omega)}{p(\mathbf{X}_k)} \tag{3}$$

The source of bias in most existing adaptive image-plane samplers is revealed here. Adaptive sampling without bias must avoid decisions to terminate sampling at an individual pixel, and instead look at the entire image plane to decide where a certain number of new samples will be cast. Every pixel with non-zero brightness must have non-zero probability of being chosen for a sample, regardless of its estimated error.

We also note the Equation 3 can be broken into many integrals, one for the support of each pixel. Provided $p(\mathbf{x})$ is known in each sub-domain, the global nature of $p(\mathbf{x})$ is not important.

4.1 The PMC-IP Kernel Function

The kernel function is the starting point in creating a PMC algorithm for adaptive image-plane sampling. We need a function that has adaptable parameters, is cheap to sample from, and supports stratification. This can be achieved with a *mixture model* of component distributions, $h_{IP,(i,j)}(\mathbf{x})$, one for each pixel:

$$K_{IP}^{(t)}(\mathbf{x}) = \sum_{(i,j)\in\mathscr{P}}\alpha_{(i,j)}^{(t)}h_{IP,(i,j)}(\mathbf{x}), \qquad \sum_{(i,j)\in\mathscr{P}}\alpha_{(i,j)}^{(t)} = 1.$$

Where (i,j) is the pixel coordinate and $\mathscr{P}$ is the set of all pixels in this image. Each component is uniform over the domain of a single pixel integral. The parameters to the distribution are all the $\alpha_{(i,j)}^{(t)}$ values, and these change for each iteration, t. We achieve an unbiased result if every $\alpha_{(i,j)}^{(t)} \geq \varepsilon$, where ε is a small positive constant (we use 0.01). We enforce this through the adaptive process, and the use of ε, rather than 0, provides some assurance that we will not overlook important contributions (referred to as *defensive sampling* [Hesterberg 1995]).

The use of a mixture as the kernel results in a *D*-kernel PMC [Douc et al. 2005a] algorithm. Sampling from such a distribution is achieved by choosing a pixel, (i,j) according to the $\alpha_{(i,j)}^{(t)}$, and then sampling from $h_{IP,(i,j)}(\mathbf{x})$. The latter can be done with a low-discrepancy sampler within each pixel, giving sub-pixel stratification. Stratification across the entire image plane can be achieved through deterministic mixture sampling, which we describe shortly. The importance function $p(\mathbf{x})$ in Equation 3 for a given pixel is $p(\mathbf{x}) = h_{IP,(i,j)}(\mathbf{x})$. This can be derived by considering each pixel as an individual integral and observing that only one mixture component has non-zero probability of contributing to each integral.

Notice that this kernel function is not conditional: $K_{IP}(\mathbf{x}^{(t)}|\mathbf{x}^{(t-1)}) = K_{IP}(\mathbf{x}^{(t)})$. Hence, for image-plane sampling we do not include a resampling step in the PMC algorithm because no samples are re-used. The knowledge gained from prior samples is instead used to adapt the kernel function.

4.2 Adapting the PMC-IP Kernel

The adaption method is responsible for determining the value of each $\alpha_{(i,j)}^{(t)}$ given the populations from previous iterations and any information available from them, such as the image computed so far. We need to to define a $\alpha_{(i,j)}^{(t)}$ for every pixel, with pixels that require more samples having higher high $\alpha_{(i,j)}^{(t)}$ for the component that covers the pixel.

An appropriate criteria assigns $\alpha_{(i,j)}^{(t)}$ proportional to the perceptually-weighted variance at each pixel. The algorithm tracks the sample variance in power seen among samples that contribute to each pixel. To account for perception, the result is divided by the threshold-versus-intensity function $tvi(L)$ introduced by Ferweda et al. [Ferwerda et al. 1996]. Normalization also accounts for ε.

$$\begin{aligned}\alpha'_{i,j} &= \frac{\overline{\sigma}^2_{(i,j)}}{tvi(L_{(i,j)})}\\ \alpha_{i,j}^{(t)} &= \varepsilon + \frac{(1-\varepsilon)\alpha'_{(i,j)}}{\sum_{(i',j')\in\mathscr{P}}\alpha'_{(i',j')}}\end{aligned}$$

The first iteration of the algorithm samples uniformly over the image plane, so this criteria can always be computed. The left images in Figure 4 show an example of an $\alpha_{(i,j)}^{(0)}$ map for a given initial image. The perceptual term in the error image prevents very high errors in both bright regions (a problem with unweighted variance) and dark areas (a problem with luminance-weighted variance).

Note that $\alpha_{(i,j)}^{(t)} \geq \varepsilon$, so there is a non-zero probabbility of generating a sample at any given image plane location. This meets the requirement for importance sampling that the importance function is non-zero everywhere where the integrand is non-zero. Furthermore, as the total sample count approaches infinity, the count at any pixel also approaches infinity. Hence, with the correctly computed importance weights (Equation 3), the algorithm is unbiased.

4.3 Deterministic Mixture Sampling

Randomly sampling from the discrete distribution defined by the $\alpha_{(i,j)}^{(t)}$ produces excess noise — some pixels get far more or fewer samples than their expected value. This problem is solved with *deterministic mixture sampling*, DMS, which is designed to give each component (pixel) a number of samples roughly proportional to its $\alpha_{(i,j)}^{(t)}$. Deterministic mixture sampling is unbiased and always gives lower variance when compared to random mixture sampling, as proven by Hesterberg [Hesterberg 1995].

The number of samples per iteration, N, (the population size) is fixed at a small multiple of the number of pixels. We typically use 4 samples per pixel, which balances between spending too much effort on any one iteration and the overhead of computing a new set of kernel parameters. For each pixel, the deterministic sampler computes $n'_{(i,j)} = N\alpha_{(i,j)}$, the target number of samples for that pixel. It takes $\lfloor n'_{(i,j)} \rfloor$ samples from each pixel (i,j)'s component. The remaining un-allocated samples are sampled from the *residual distribution* with probability $n'_{(i,j)} - \lfloor n'_{(i,j)} \rfloor$ at each pixel (suitably normalized).

Figure 3 summarizes the final PMC-IP algorithm:

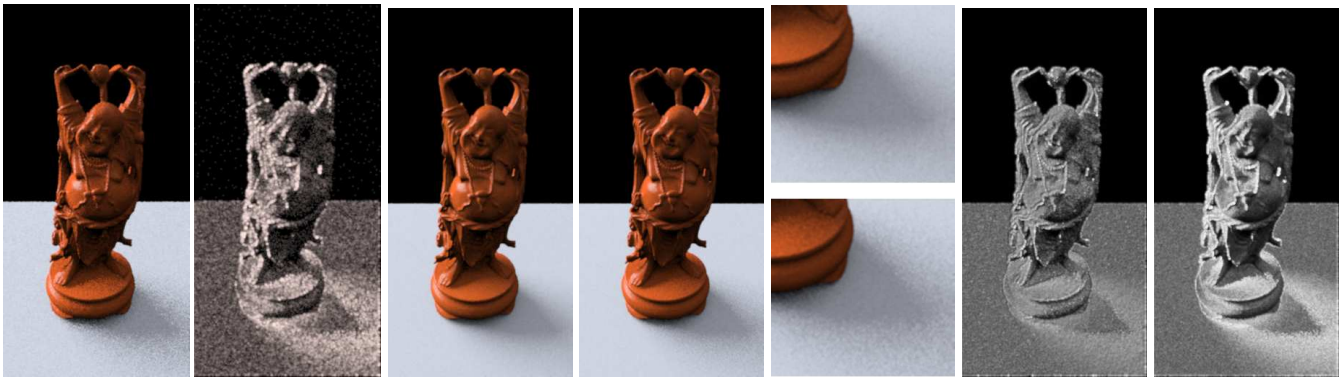

Figure 4: A comparison between adaptive and uniform image-plane sampling on a direct lighting example. Leftmost is the initial image for PMC-IP sampling, and the $\alpha_k^{(0)}$ image. The initial image used 2 samples per pixel. The next image is the result of PMC-IP sampling with two iterations at 4spp on average. Center is a 10spp image uniformly distributed. The zooms show the shadow near the Buddha's base (PMC-IP top, uniform bottom). To the right are the corresponding variance images. Note that the variance image for the PMC-IP sampler has few high variance regions, and has a lower contrast in general, representing a more even distribution of error.

1 Generate the initial image
2 **for** $t = 1, \cdots, T$
3 Compute the perceptually-weighted variance image
4 Compute $\alpha_k^{(t)}$ for each pixel k
5 Use DMS to allocate samples according to $\alpha_k^{(t)}$
6 Generate samples from $K_{IP}^{(t)}(\mathbf{x})$ and accumulate to image

Figure 3: The PMC-IP Algorithm.

Image	Method	# SPP	T(s)	Err	P-Eff
Buddha	Uniform	10	58.1	0.625	0.027
	PMC-IP	2+4+4	62.4	0.116	0.138
Box	Uniform	16	163	0.545	0.011
	Uniform	32	328	0.255	0.012
	PMC-IP	4+6+6	169	0.182	0.033

Table 1: Measurements comparing PMC-IP and uniform image-plane sampling, for equal total sample counts. The Buddha image computed direct lighting with the MIS method, with a total of 8 lighting samples for each pixel sample. PMC-IP sampling improves the perceptual-based RMS error by a factor of 5.4 over uniform sampling with only 7.5% more computation time. It corresponds to an improvement in efficiency of 5.01. The Cornell Box images use path tracing to compute global illumination including caustics. Comparing with images of 16ssp, PMC-IP improves the efficiency by a factor of 2.65.

4.4 PMC-IP Results

Adaptive image-plane sampling can be used in many situations where pixel samples are required and an iterative algorithm can be employed. We have implemented it in the contexts of direct lighting using a Multiple Importance Sampler (MIS) and global illumination with path tracing, and as part of a complete photon mapping system, which we discuss in Section 6.

Figure 4 shows the Buddha direct lighting example. The surface is diffuse with an area light source. Each pixel sample used 8 illumination samples, and the images were rendered at 256×512, with statistics presented in Table 1. We introduce the perceptually-based mean squared efficiency (P-Eff) metric for comparing algorithms, computed as:

$$Err = \frac{\sum_{pixels} e^2}{tvi(L)}, \quad P\text{-}Eff = \frac{1}{T \times Err}$$

where e is the difference in intensity between a pixel and the ground truth value and T is the running time of the algorithm on that image. P-Eff is a measure of how much longer (or less) you would need to run one algorithm to reach the perceptual quality of another [Pharr and Humphreys 2004].

The final adaptive image shown is the unweighted average of three sub-images (initial and two iterations). While weighting each sub-image may be helpful, in this context it is not clear that the samples from one iteration are any better than those from another because they all used the same per-sample parameters. We obtained more samples in places that needed it, but not better samples.

The path tracing algorithm differs from a standard version only in how pixel locations are chosen. The improvement due to PMC-IP sampling is more pronounced in this situation because some areas of the image (the caustic, for instance) have a much higher variance than others due to the difficulty of sampling such paths. We compare the results in two aspects. First, we compare them visually. Working toward a target image quality, we would continue iterating the PMC-IP sampler until we were satisfied with the overall variance. In Fig. 5, we show the final result of the Cornell box and the comparison between a set of snapshots of the caustic region between the general PT algorithm and our adaptive algorithm. We can see that the result of 16th (equivalent to 64 spps) is even better than the result of 256 spps. We also notice that even at diffuse regions, our method converges more quickly than the general PT algorithm.

Second, we compare the efficiencies of our algorithm and the PT algorithm. In this Table 1, we see that PMC-IP sampling with a total of 16spp improves the efficiency by a factor of 3 to the uniform sampling with 16 spps and 32 spps. In this result, we ran our examples for a fixed number of iterations (bounded by computation time). Note that because the PMC-IP sampler evenly spreads variance over the image, an overall image error bound is very unlikely to leave any high-error pixels.

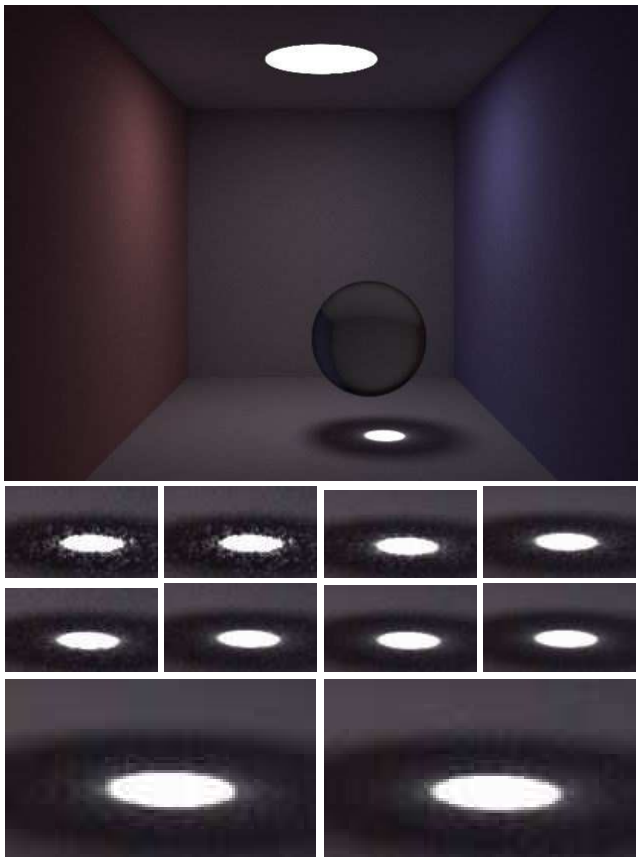

Figure 5: A Cornell Box image computed using the PMC-IP algorithm. The top image is the final result using PMC-IP algorithm with 64 iterations, with each iteration averaging 4 spps. It is easier to find converged values in diffuse regions than in the caustic region. Thus, we compare the results by focusing on this region. The images in the second row, from left to right, are the cropped images of the caustic region computed using non-adaptive path tracing with 16, 32, 64 and 128 spps. The images in the third row, from left to right, are intermediate results from the adaptive algorithm at 4, 8, 16 and 32 iterations when computing the top image. The last row demonstrates that our adaptive sampler produces better visual results at lower sample counts: on the left is the result from 256 spps, un-adapted, and on the right image is the result of 16 adapting iterations at an average of 4 spps per iteration.

5 PMC-HI: Adaptive Hemispheric Integrals Sampling

Hemispheric samplers generate incoming directions, ω', at a surface point, $\mathbf{x}$. One application is in direct lighting, which assumes that the light leaving a surface point, $L(\mathbf{x},\omega)$ can be evaluated by the following integral, composed of terms for light emitted from and reflected at $\mathbf{x}$:

$$L(\mathbf{x},\omega) = L_e(\mathbf{x},\omega) + \int_\Omega f(\mathbf{x},\omega,\omega')d\omega' \tag{4}$$

where $L_e(\mathbf{x},\omega)$ is light emitted at $\mathbf{x}$, Ω is the hemisphere of directions *out* of $\mathbf{x}$ and $f(\mathbf{x},\omega,\omega')$ is the light reflected at $\mathbf{x}$ from direction $-\omega'$ into direction ω:

$$f(\mathbf{x},\omega,\omega') = L_{in}(\mathbf{x},-\omega')\,f_r(\mathbf{x},\omega,\omega')|\cos(\theta')| \tag{5}$$

where $L(\mathbf{x},-\omega')$ is the light arriving at $\mathbf{x}$ from direction ω', $f_r(\mathbf{x},\omega,\omega')$ is the BRDF, and θ' is the angle between ω' and the normal at $\mathbf{x}$.

A standard importance sampling algorithm for $L(\mathbf{x},\omega)$ samples directions, $\{\omega'_1,\ldots,\omega'_n\}$, out of $\mathbf{x}$ according to an importance function, p, and computes the estimate:

$$\hat{L}(\mathbf{x},\omega) = \frac{1}{n}\sum_{i=1}^{n}\frac{f(\mathbf{x},\omega,\omega'_i)}{p(\omega'_i)} \tag{6}$$

The variance of this estimator improves as p more closely approximates f, and is zero when p is proportional to f.

In the local direct lighting situation, one common choice for p is proportional to $L_{in}(\mathbf{x},-\omega')\,f_r(\mathbf{x},\omega,\omega')|\cos(\theta')|$ or a normalized approximation to it. An alternative is to break the integral into a sum over individual light sources and sample points on the lights to generate directions [Pharr and Humphreys 2004, §16.1]. In an environment map lighting situation, the wavelet product approach of Clarberg et al. [2005] currently provides the best way to choose p. However, none of these individual importance functions behaves well in all cases.

Figure 6 demonstrates the various difficult cases for importance sampling. The floor consists of a checker pattern with diffuse and glossy squares (with two types of gloss settings). There are two lights, one large and one small. In pixels that image diffuse squares, an importance function based on the lights is best. In highly glossy pixels that reflect the large light, BRDF sampling is best. For glossy pixels that do not reflect light, sampling from the light is best, and rough glossy pixels benefit from both BRDF and light sampling, but we have no way of knowing this a-priori, and most practitioners would use BRDF sampling. In rough glossy regions that reflect only one light, sampling from the other light is wasteful, but again most algorithms would sample equally or according to total emitted power.

Multiple Importance Sampling (MIS) and Bidirectional Importance Sampling address many of these problems, by trying several importance functions and combining their results. While this does very well at reducing variance, it is wasteful in cases where one of the importance functions is much better than the others and could be used alone. Other techniques assume knowledge of which strategy will dominate where.

PMC-HI is a sampler that generates directions out of a point by adapting a kernel function to match the integrand of interest —

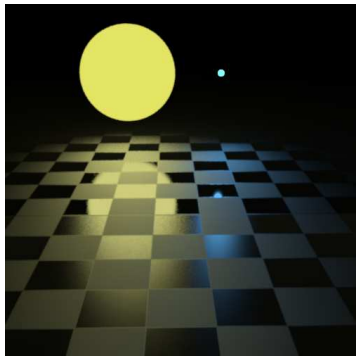

Figure 6: A scene constructed to demonstrate how the optimal sampling strategy varies over an image. The checkers contains diffuse and glossy squares, with near-pure specular toward the back and rougher toward the front. There are two light sources.

$L_{in}(\mathbf{x}, -\omega')\, f_r(\mathbf{x}, \omega, \omega')|\cos(\theta')|$ in the direct lighting case. For example, the lower images in Figure 7 indicate the relative usefulness of different importance functions at each pixel. Furthermore, the PMC framework enables important samples from one iteration to guide sampling in subsequent iterations.

5.1 The PMC-HI Kernel Function

Each direct lighting estimate takes place at a single surface point and is only one small step in a larger computation. The same surface point, and hence the same target function, f_r, essentially never reappears. We choose to adapt on a per-estimate basis, which avoids the need to store information about the adaptation state at the surface points and interpolate to find information at new points. Hence, the number of samples on which to base adaption is low, certainly less than 100 and less than 10 in some of our examples.

A mixture distribution of a few candidate importance functions is a good starting point. At least one such component is likely to be a good approximation to f_r, and we expect to adapt to use that function most often. To catch cases where good sampling directions are hard to find, we include a component, h_{cone}, that samples based on important sample directions from the previous iteration. For one light, the mixture is

$$\begin{aligned} K_{IR}^{(t)}(\omega^{(t)}|\mathbf{d}^{(t)}, \beta^{(t)}) &= \alpha_{BRDF}^{(t)} h_{BRDF}(\omega^{(t)}) \\ &\quad + \alpha_{light}^{(t)} h_{light}(\omega^{(t)}) \\ &\quad + \alpha_{cone}^{(t)} h_{cone}(\omega^{(t)}|\mathbf{d}^{(t)}, \beta^{(t)}) \end{aligned} \tag{7}$$

There is one term for the BRDF-based importance function, one for a light (or one per light for multiple lights) and the cone perturbation function. The cone function samples a direction uniformly within a cone of directions with axis $\mathbf{d}^{(t)}$ and half-angle $\beta^{(t)}$, which is set based on the population in the previous iteration. It is particularly useful for situations like partial shadowing where previous samples that found visible portions of the light generate more samples that also reach the light.

The population in PMC-HI is a set of sample directions out of the surface point we are estimating. The population size must be large enough to obtain reasonable estimates for the $\alpha_k^{(t)}$ values at each iteration but not so large as to increase computation times unnecessarily. We typically use $N = 2m$, where m is the number of mixture components. This is a sufficient size to see the benefits of adaption, as the results in Figure 7 demonstrate.

5.2 Adapting for PMC-HI

An initial population of N samples, $\left\{\Omega_1^{(0)}, \ldots, \Omega_{n_0}^{(0)}\right\}$, is generated using $\alpha_{cone}^{(0)} = 0$ and the other $\alpha_k^{(0)}$ equal and summing to one. A deterministic mixture sampling is used to select the number of samples from each component. Each sample is tagged with the mixture component that was used to generate it, and their importance weights are computed:

$$w_i^{(0)} = \frac{f(\mathbf{x}, \omega, \omega')}{K_{IR}^{(0)}(\omega^{(0)})} \tag{8}$$

There is no resampling step for direct lighting. The sample size is so small that resampling tends to unduly favor high-weight directions at the expense of others, thus reducing the degree to which sampling explores the domain. Instead, the cone mixture component is used to incorporate the information from previous samples.

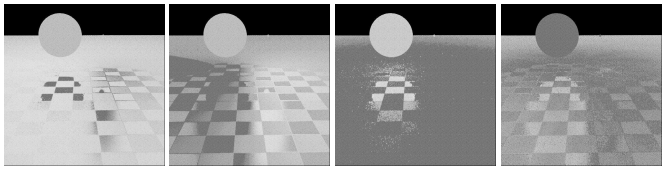

Figure 7: These maps show how the mixture component weights for PMC-HI vary over the image, after two iterations. Bright means high weight. From left to right: $\alpha_{L1}^{(2)}$, the left light's weight; $\alpha_{L2}^{(2)}$, the right light's weight; $\alpha_{BRDF}^{(2)}$; and $\alpha_{cone}^{(2)}$, which in this image is of limited use. The large light dominates in regions where no light is seen in a glossy reflection, while the right light is favored in nearby diffuse squares. The BRDF component is favored only when the large light is specularly reflected at a pixel. The images are quite noise-free for such small sample counts (16 total samples per estimate), indicating that the adaption mechanism converges to a consistent result.

Image	Method	# SPP	T(s)	Err	P-Eff
Checks	MIS	12	46	0.379	0.057
	MIS	48	183	0.153	0.035
	PMC-HI	12	54	0.146	0.127
Plant	MIS	27	53	0.403	0.047
	PMC-HI	27	64	0.128	0.122

Table 2: Measurements comparing PMC-HI sampling with MIS, for equal total sample counts. In all cases we used a single direct lighting estimate for each pixel. For the Checks scene, PMC-HI improve the efficiency by a factor 2.21, which takes four times more samples for uniform MIS to reach the approximately same perceptual based variance (Err). The efficiency gain for the Plant scene is 2.60.

The new component weights, $\alpha_k^{(1)}$, can now be determined, along with the $\mathbf{d}^{(1)}$ and $\beta^{(1)}$ parameters for $h_{cone}(\omega^{(1)}|\mathbf{d}^{(1)}, \beta^{(1)})$. The cone direction $\mathbf{d}^{(1)}$ is found by taking a weighted average of the $t = 0$ population samples, with weights $w_i^{(0)}$. The cone size is set to the standard deviation of those samples. The component weights are set based on the sample importance weights:

$$\alpha_k^{(t)} = \frac{\sum_{i \in \mathscr{S}_k} w_i^{(t-1)}}{\sum_{j=1}^{n} w_j^{(t-1)}} \tag{9}$$

where $\mathscr{S}_k$ is the set of samples that were generated using component k. In the first iteration there is no sample from the cone perturbation, so we set $\alpha_{cone}^{(1)} = 0.2$ and adjust the other α's by a factor of 0.8 to make them all sum to one.

We now begin the next iteration. A new set of samples is generated using deterministic mixture sampling from the kernel $K_{IR}^{(t)}(\omega^{(t)}|\mathbf{d}^{(t)}, \beta^{(t)})$, weights are computed, and the kernel function is updated based on the weights. To form the estimate, we use Equation 1, with each sample, $\Omega_i^{(t)}$, weighted by $w_i^{(t)}$ from Equation 8.

5.3 Adaptive Direct Lighting Results

We present results on two examples of PMC-HI for direct lighting: the checker scene (Figure 6) and a plant rendering with complex

Figure 8: An image involving complex soft shadows and glossy surfaces. The top is PMC-HI sampling, while the middle is MIS with equal total sample count. Note the significant improvement in the soft shadows achieved with PMC-HI, shown in the zoomed images at the bottom (PMC-HI left, MIS right).

shadows and glossy BRDFs (Figure 8). The timing and the error comparisons with MIS (the best of several existing algorithms we tried on these scenes) appear in Table 2. The checkers image resolution is 500×500 and the plant image is at 720×405.

The checker scene clearly demonstrates that adaption is a stable process that finds a good kernel function, or evenly weights the components if no component dominates (Figure 7). The cone component is not particularly helpful in this case because the visibility is simple. The results show that PMC-HI gains an improvement in rendering efficiency by a factor of about 3. The plant scene demonstrates the usefulness of the cone function in partially shadowed regions. It results in a major improvement in the soft shadow boundaries on the table.

6 Integrating Samplers into a Rendering System

The combined use of both PMC-IP and PMC-HI in a single rendering pipeline allows us to further improve the rendering efficiency. Figure 9 shows a modern plug-in style Monte Carlo rendering framework. The only core framework modification required to support adaptive sampling is the addition of a feedback path from the output image generator back to the samplers, required to pass information from one sampling iteration back to the samplers for the next iteration. For the PMC-IP sampler, this feedback provides the variance map required to determine pixel sampling weights. In addition, we can improve the performance of the PMC-HI sampler through feedback.

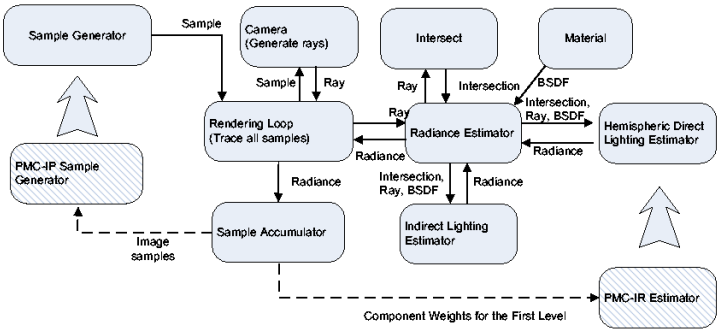

Figure 9: A block diagram of a plug-in style Monte Carlo rendering system, following Pharr and Humphreys [2004]. The PMC-IP sampler replaces a uniform sample generator with the addition of a feedback path from the sample accumulator in order to calculate the perceptual variance. The PMC-HI estimator replaces the direct lighting estimator. We can feed information from one iteration back to the next one to provide initial values for the next iteration.

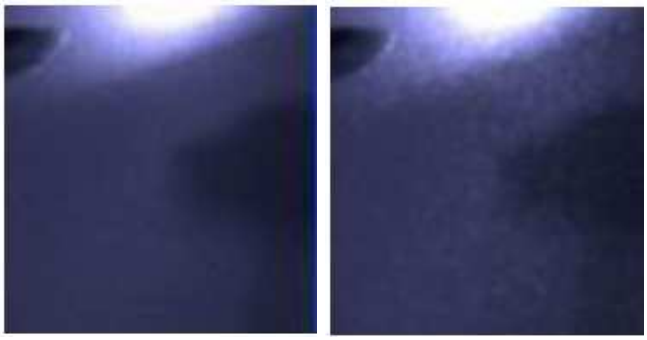

Figure 10: Blown up images of the upper right portion of the room scene from Figure 1. The image was generated with a standard photon shooting phase. On the left is the result of a final gather with 4 PMC-IP iterations, with each iteration averaging 4 samples per pixel and using the PMC-HI direct lighting sampler augmented with the feedback mechanism. PMC-HI uses 2 iterations and each iterations has 16 shadow rays to estimate direct lighting. Right is the result of a standard photon mapping gather using 16 spps and using 16 shadow rays per light to estimate direct lighting. Note the significant reducitomn in noise with our methods.

6.1 Improving the PMC-HI with Feedback

In Figure 9 we show the PMC-HI estimator replacing the direct lighting estimator. However, it can be used in any situation where the estimation of an integral over the hemisphere is required. Irradiance caching would benefit from the PMC-HI estimator in the computation of each cache value. Photon mapping can also use a PMC sampler in the final gathering phase.

The PMC-HI sampler of Section 5 was presented in the context of a single integral estimate. On the first iteration it uses a set of default weights for each mixture component, $\alpha_k^{(0)}$. In a complete rendering system, we are able to record the final adapted weights from all the direct lighting integrals done during one image-plane sampling iteration. The recorded weights can then be used to initialize the PMC-HI $\alpha_k^{(t)}$ values for the next round of sampling.

Consider a single pixel sample at image plane iteration t. A direct lighting estimate is made for the first surface point seen through the pixel location. This direct lighting estimate uses the PMC-HI sampler to adapt a set of mixture weights, which we pass back along with the estimate itself and accumulate in an auxiliary image. Other estimates may also be made for indirect illumination, but we do not record the weights from those.

At the next image-plane iteration, any direct lighting estimate required at a pixel looks to the auxiliary image to find starting weights for the PMC-HI adaption process. This avoids wasting PMC-HI iterations with un-adapted initial parameters, which improves our rendering efficiency.

Photon mapping is an industry standard method for global illumination, and we implemented the above method for the gather portion of a photon mapping implementation. Figure 1 shows a room scene computed with the final system. Looking at the blown-up images of right wall by the lamp, in Figure 10, we can see that our algorithm converges more rapidly to a smooth image. This is because PMC-HI obtains a better estimate of the direct lighting, and PMC-IP puts more samples in this region because of its high variance nature. Both improve the efficiency of the final result.

7 Discussion

The most important variable parameter in a PMC algorithm is the population size. Assuming a fixed total rendering budget, a small population reduces the number of samples per iteration, which gives more flexibility in the total sample count in an algorithm, but relatively more time is then spent adapting mixture parameters. Furthermore, the quality of the adapted functions is lower because they are derived from less information. Hence, we use small populations only for the hemispheric integrals case, where we aim to keep the total number of samples per estimate low and the kernel function has a very small number of parameters. Larger populations result in more robust adaptation and less overhead, and in general are to be favored. However, if the population is too large, the benefits of adaption are lost as relatively more samples are drawn using a mal-adapted importance function during the early iterations.

In Equation 8 we use the full mixture distribution as the importance function, $K(\omega_i')$. This is a form of Rao-Blackwellization, which reduces variance but at the expense of additional computation. The algorithm remains correct if we use only the mixture component from which the sample came, $h_k(\omega'^{(i)})$, and we need not compute the other mixture functions. In some cases, the resulting reduction in computation may exceed the increase in noise, but in rendering the greatest cost is usually in obtaining a sample, rather than evaluating its probabilities.

The most notable limitation of PMC is the high sample counts required when the kernel has many adaptable parameters. This precludes, for instance, using one component per light when there are many lights. Such a strategy would be appealing for efficiently sampling in complex shadow situations (some components would see the lights, others would not), but the sample count required to adequately determine the mixture component weights would be too large. Instead we use a single mixture component for all the lights and rely on the cone perturbation component to favor visible lights. This does not work well if the illumination sources are widely spaced.

In Section 6 we presented an image space method for sharing adapted parameters across different estimates. An alternate approach for integrating functions defined on surfaces is to store the mixture component weights in a surface map and interpolate. This amortizes the cost of adapting over many surface points. We did not explore this possibility, but it offers potential for the multi-light problem or cases where many light transport paths must be constructed through a scene, such as bi-directional path tracing or photon mapping.

8 Conclusion

We have shown how algorithms for adaptive image-plane sampling and hemispheric integral computations can be derived within a PMC framework. In each case the algorithm learns an effective sampler based on the results from early iterations. This alleviates one of the greatest problems in Monte Carlo rendering: the choice of importance functions and other parameters.

The image-plane sampler and direct lighting integrator are common components in many rendering algorithms. PMC-IP sampling could be used as a plugin component for essentially any algorithm that forms light paths through the eye, including the gather phase of photon-mapping, bi-directional path tracing, irradiance caching, and so on. The PMC-HI sampler could be used in any situation where estimates of an integral over the hemisphere are required. Irradiance caching would benefit greatly from a PMC sampler in the computation of each cached value. We have shown how photon mapping can use PMC samplers in the final gathering phase.

PMC is just one approach from the family of iterated importance sampling algorithms [Robert and Casella 2004]. The Kalman filter is another well-known example. Common to these techniques is the idea of sample reuse through resampling and the adaption of sampling parameters over iterations. Computer graphics certainly offers further opportunities to exploit these properties.

References

AGARWAL, S., RAMAMOORTHI, R., BELONGIE, S., AND JENSEN, H. W. 2003. Structured importance sampling of environment maps. In *SIGGRAPH '03*, 605–612.

ANONYMOUS. 2006. *Sequential Monte Carlo Methods for Physically based rendering.* PhD thesis, Anonymous University.

BALÁZS, B., SZIRMAY-KALOS, L., AND GYÖRGY, A. 2003. Weighted importance sampling in shooting algorithms. In *Proc. of the Spring Conference on Computer Graphics*, 177–184.

BOLIN, M. R., AND MEYER, G. W. 1998. A perceptually based adaptive sampling algorithm. In *SIGGRAPH '98*, 299–309.

BURKE, D., GHOSH, A., AND HEIDRICH, W. 2005. Bidirectional importance sampling for direct illumination. In *Proc. of the 16th Eurographics Symposium on Rendering*, 147–156.

CAPPÉ, O., GUILLIN, A., MARIN, J.-M., AND ROBERT, C. 2004. Population Monte Carlo. *Journal of Computational and Graphical Statistics 13*, 4, 907–929.

CLARBERG, P., JAROSZ, W., AKENINE-MÖLLER, T., AND JENSEN, H. W. 2005. Wavelet importance sampling: efficiently evaluating products of complex functions. In *SIGGRAPH '05*, 1166–1175.

CLINE, D., TALBOT, J., AND EGBERT, P. 2005. Energy redistribution path tracing. In *SIGGRAPH '05*, 1186–1195.

CSONKA, F., SZIRMAY-KALOS, L., AND ANTAL, G. 2002. Generalized multiple importance sampling for monte-carlo global illumination. *Machine Graphics & Vision International Journal archive 11*, 3–20.

DAYAL, A., WOOLLEY, C., WATSON, B., AND LUEBKE, D. 2005. Adaptive frameless rendering. In *Proc. of the 16th Eurographics Symposium on Rendering*, 265–275.

DIPPÉ, M. A. Z., AND WOLD, E. H. 1985. Antialiasing through stochastic sampling. In *SIGGRAPH '85*, 69–78.

DOUC, R., GUILLIN, A., MARIN, J. M., AND ROBERT, C. P. 2005. Convergence of adaptive sampling schemes. Technical Report 2005-6, University Paris Dauphine.

DOUC, R., GUILLIN, A., MARIN, J. M., AND ROBERT, C. P. 2005. Minimum variance importance sampling via population Monte Carlo. Technical report, University Paris Dauphine.

DUTRÉ, P., AND WILLEMS, Y. D. 1994. Importance-driven Monte Carlo light tracing. In *Proc. of the 5th Eurographics Workshop on Rendering*, 185–194.

DUTRÉ, P., AND WILLEMS, Y. D. 1995. Potential-driven Monte Carlo particle tracing for diffuse environments with adaptive probability functions. In *Proc. of the 6th Eurographics Workshop on Rendering*, 306–315.

FAN, S., CHENNEY, S., AND LAI, Y. 2005. Metropolis photon sampling with optional user guidance. In *Proc. of the 16th Eurographics Symposium on Rendering*, Eurographics Association, 127–138.

FARRUGIA, J.-P., AND PÉROCHE, B. 2004. A progressive rendering algorithm using an adaptive perceptually based image metric. *Computer Graphics Forum (Proc. of Eurographics 2004) 23*, 3, 605–614.

FERWERDA, J. A., PATTANAIK, S. N., SHIRLEY, P., AND GREENBERG, D. P. 1996. A model of visual adaptation for realistic image synthesis. In *SIGGRAPH '96*, 249–258.

GHOSH, A., DOUCET, A., AND HEIDRICH, W. 2006. Sequential sampling for dynamic environment map illumination. In *Proc. Eurographics Symposium on Rendering*, 115–126.

GLASSNER, A. 1995. *Principles of Digital Image Synthesis*. Morgan Kaufmann.

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

KIRK, D., AND ARVO, J. 1991. Unbiased sampling techniques for image synthesis. In *SIGGRAPH '91*, 153–156.

LAFORTUNE, E. P., AND WILLEMS, Y. D. 1994. The ambient term as a variance reducing technique for Monte Carlo ray tracing. In *Proc. of the 5th Eurographics Workshop on Rendering*, 168–176.

LAFORTUNE, E. P., AND WILLEMS, Y. D. 1995. A 5D tree to reduce the variance of Monte Carlo ray tracing. In *Proc. of the 6th Eurographics Workshop on Rendering*, 11–20.

LEE, M. E., REDNER, R. A., AND USELTON, S. P. 1985. Statistically optimized sampling for distributed ray tracing. In *SIGGRAPH '85*, 61–68.

MITCHELL, D. P. 1987. Generating antialiased images at low sampling densities. In *SIGGRAPH '87*, 65–72.

NEILSON, D., AND YANG, Y.-H. 2003. Variance invariant adaptive temporal supersampling for motion blurring. In *PG '03: Proc. of the 11th Pacific Conference on Computer Graphics and Applications*, 67–72.

OHBUCHI, R., AND AONO, M. 1996. Quasi-Monte Carlo rendering with adaptive sampling. Technical Report RT0167, IBM Tokyo Research Laboratory.

OWEN, A., AND ZHOU, Y. 2000. Safe and effective importance sampling. *Journal of the American Statistical Association 95*, 135–143.

PAINTER, J., AND SLOAN, K. 1989. Antialiased ray tracing by adaptive progressive refinement. In *SIGGRAPH '89*, 281–288.

PHARR, M., AND HUMPHREYS, G. 2004. *Physically Based Rendering from Theory to Implementation*. Morgan Kaufmann.

PIETREK, G., AND PETER, I. 1999. Adaptive wavelet densities for Monte Carlo ray tracing. In *WSCG'99 Conference Proceedings*, V. Skala, Ed., 217–224.

PURGATHOFER, W. 1986. A statistical method for adaptive stochastic sampling. In *Proc. EUROGRAPHICS 86*, 145–152.

RAMASUBRAMANIAN, M., PATTANAIK, S. N., AND GREENBERG, D. P. 1999. A perceptually based physical error metric for realistic image synthesis. In *SIGGRAPH '99*, 73–82.

RIGAU, J., FEIXAS, M., AND SBERT, M. 2002. New contrast measures for pixel supersampling. In *Proc. of CGI'02*, Springer-Verlag, 439–451.

RIGAU, J., FEIXAS, M., AND SBERT, M. 2003. Entropy-based adaptive sampling. In *Proc. of Graphics Interface 2003*, 149–157.

ROBERT, C. P., AND CASELLA, G. 2004. *Monte Carlo Statistical Methods*, 2nd ed. Springer-Verlag.

SCHLICK, C. 1991. An adaptive sampling technique for multidimensional integration by ray-tracing. In *Proc. of the 2nd Eurographics Workshop on Rendering*, 21–29.

SHIRLEY, P., WANG, C., AND ZIMMERMAN, K. 1996. Monte Carlo techniques for direct lighting calculations. *ACM Transactions on Graphics 15*, 1 (Jan), 1–36.

SILLION, F., AND PUECH, C. 1994. *Radiosity and Global Illumination*. Morgan Kaufmann.

STOKES, W. A., FERWERDA, J. A., WALTER, B., AND GREENBERG, D. P. 2004. Perceptual illumination components: a new approach to efficient, high quality global illumination rendering. In *SIGGRAPH '04*, 742–749.

SZÉCSI, L., SBERT, M., AND SZIRMAY-KALOS, L. 2004. Combined correlated and importance sampling in direct light source computation and environment mapping. *Computer Graphics Forum (Proc. of Eurographics 2004) 23*, 3, 585–593.

TALBOT, J., CLINE, D., AND EGBERT, P. 2005. Importance resampling for global illumination. In *Proc. of the 16th Eurographics Symposium on Rendering*, 139–146.

TAMSTORF, R., AND JENSEN, H. W. 1997. Adaptive sampling and bias estimation in path tracing. In *Proc. of the 8th Eurographics Workshop on Rendering*, 285–296.

VEACH, E., AND GUIBAS, L. J. 1995. Optimally combining sampling techniques for Monte Carlo rendering. In *SIGGRAPH '95*, 419–428.

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

WARD, G. 1991. Adaptive shadow testing for ray tracing. In *Proc. of the 2nd Eurographics Workshop on Rendering*, 11–20.