COMPUTER GRAPHICS forum

DOI: 10.1111/j.1467-8659.2010.01713.x

COMPUTER GRAPHICS forum
Volume 00 (2010), number 0 pp. 1–11

Arbitrary Importance Functions for Metropolis Light Transport

Jared Hoberock* and John C. Hart

University of Illinois Urbana-Champaign, USA
jhoberock@nvidia.com, jch@illinois.edu

Abstract

We present a generalization of the scalar importance function employed by Metropolis Light Transport (MLT) and related Markov chain rendering algorithms. Although MLT is known for its user-designable mutation rules, we demonstrate that its scalar contribution function is similarly programmable in an unbiased manner. Normally, MLT samples light paths with a tendency proportional to their brightness. For a range of scenes, we demonstrate that this importance function is undesirable and leads to poor sampling behaviour. Instead, we argue that simple user-designable importance functions can concentrate work in transport effects of interest and increase estimator efficiency. Unlike mutation rules, these functions are not encumbered with the calculation of transitional probabilities. We introduce alternative importance functions, which encourage the Markov chain to aggressively pursue sampling goals of interest to the user. In addition, we prove that these importance functions may adapt over the course of a render in an unbiased fashion. To that end, we introduce multi-stage MLT, a general rendering setting for creating such adaptive functions. This allows us to create a noise-sensitive MLT renderer whose importance function explicitly targets noise. Finally, we demonstrate that our techniques are compatible with existing Markov chain rendering algorithms and significantly improve their visual efficiency.

Keywords: rendering, light transport, sampling, Metropolis light transport

ACM CCS: 1.3.7 [Computer Graphics] Raytracing

1. Introduction

Monte Carlo integration is the foundation of a large class of Physically based photorealistic rendering algorithms. It estimates the integral of a function $f(x)$ over some domain Ω as the sum of N random samples X_i distributed over Ω with probability density $p(x)$,

$$\int_\Omega f(x)\mathrm{d}\mu(x) \approx \frac{1}{N}\sum_{i=1}^{N}\frac{f(X_i)}{p(X_i)}. \qquad (1)$$

The problem with Monte Carlo techniques is that the sample randomness appears as noise in the resulting image, and this noise is commonly measured by the variance of the result $V[f/p] = E[(f/p)^2] - E[f/p]^2$. Importance sampling reduces this variance by setting $p(x) \approx f(x)/b$, where b is an estimate of the integral (1) needed to ensure the p.d.f. sums to one. Any p.d.f. (importance function) can be used, but (strict) importance sampling selects samples proportional to the magnitude of f, which best reduces variance.

For light transport, where $f(x)$ is the radiance of light along path x, importance sampling causes the Monte Carlo approximation of the integral to focus on high-radiance paths, because the accuracy of these paths has a greater effect on variance than does the accuracy of the low radiance paths, due to the squaring in the variance equation. The consequence of Monte Carlo importance sampled rendering is that darker areas of an image appear noisier than brighter areas.

Importance sampling is engineered to optimize image variance, but image variance is not an accurate model of the

*Currently at NVIDIA Corp.

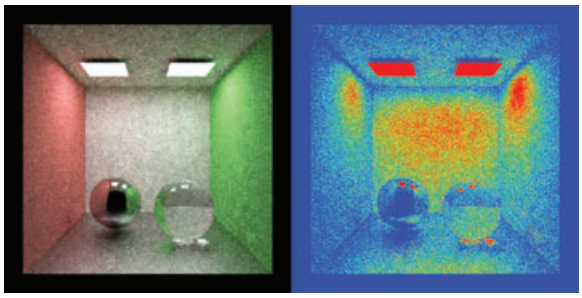


Figure 1: *Left panel: MLT in the Cornell box. Right panel: Sampling density. Most work is spent sampling the bright lights while little is left for the rest of the scene.*

perceptual fidelity of the human visual system (HVS), which is more sensitive to relative contrast than to absolute luminance. Contrast is measured by a ratio of the range of the luminance to its average, which means that the HVS tolerates a greater degree of noise in brighter areas of an image than in darker areas. It is thus valuable to explore alternative 'importance' functions designed to reduce noise more uniformly across an image.

Most popular Monte Carlo rendering methods, for example path tracing [Kaj86], bi-directional ray tracing [LW93] and photon mapping [JC95], are easily adjusted to more evenly distribute samples across an image. Metropolis light transport (MLT) is better than these at rendering pathologically difficult lighting configurations [VG97], but its mutation rules make sample distribution much more difficult.

Because its samples are perturbations of high-radiance paths, and the screen distribution of these samples is at the mercy of path mutation, MLT notoriously undersamples darker image regions, by as much as 1000:1 in the Cornell box in Figure 1. Furthermore, when a bright spot is found the mutational acceptance probability is reduced, which results in a large number of identical samples deposited at the same image pixel, yielding the bright spots of noise observed in Figure 1.

We seek to encourage stratification across image pixels and reduce the contrast sensitive perception of the resulting image noise even though this may increase the resulting image's absolute variance. We show that the MLT kernel lends itself to arbitrary user-defined importance functions that can result in stratification and noise reduction. Section 4 presents such importance functions and introduces a novel multistage adaptive MLT algorithm that includes a novel method for estimating the error of an ongoing MLT process and show how an importance function can use the information to increase sample density in areas of high image noise.

2. Previous Work

Some have examined different mutation schemes to stratify MLT samples but require a lot of effort to ensure they do not introduce bias. For example, Kelemen *et al*. [KSKAC02] devised a rule to account for a wide range of common effects, Fan *et al.* [FCL05] designed a rule to guide the sampler to user-specified light paths, and Segovia *et al.* [SIP07] extended MLT to instant radiosity. Many path mutation techniques require careful analysis and computation of transitional probability densities to ensure unbiasedness. We instead explore different importance functions which by formulation do not introduce bias.

Energy redistribution path tracing (ERPT) [CTE05] stratifies MLT samples more evenly across the screen by performing an independent MLT integration for each pixel initialized with an MC path tracing through that pixel. This approach succeeds at stratification but at the cost of two significant drawbacks. First, the decomposition of MLT's full-length Markov chain into shorter per-pixel chains inhibits MLT's ability to deeply search and thoroughly explore obscure light transport paths. Secondly, an ERPT pixel's path will eventually mutate through a different pixel, so the effect of ERPT's stratification decreases as its number of samples increases. The consequences of this trade-off is illustrated in Figure 2.

Lai *et al.* [LFCD07] enhanced ERPT using population Monte Carlo techniques to iteratively resample from an initial population of light paths and adapt various parameters of

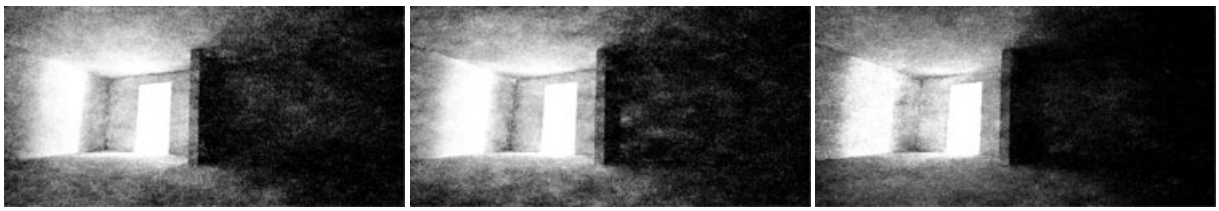


Figure 2: *MLT, ERPT and noise-aware MLT in a dark room. By utilizing an arbitrary importance function which targets visual noise, our noise-aware MLT algorithm can produce a smoother, more visually pleasing image using the same number of samples as previous methods.*

Algorithm 1: MLT with importance

1 Estimate normalization constant $b \approx \int I(x)\mathrm{d}x$
2 $x =$ initial path
3 **for** $1 \ldots N$ **do**
4 $y =$ mutate(x) chosen using $T(x \to y)$
5 Compute accept probability $a = \min\left(\frac{I(y)T(y \to x)}{I(x)T(x \to y)}, 1\right)$
6 pixel(x) $+= \frac{1-a}{N} \frac{f(x)}{I(x)/b}$
7 pixel(y) $+= \frac{a}{N} \frac{f(y)}{I(y)/b}$
8 **if** *uniformRandom()* $< a$ **then** $x = y$

the proposal density (i.e. the mutations themselves) over the course of the rendering process. By contrast, our work controls the acceptance density by manipulating the importance function.

3. MLT with Arbitrary Importance Function

Our enhancement of MLT to better stratify its samples and reduce noise is implemented as a generalized importance function. The original MLT formulations include a luminance-based importance function, but alternative importance functions can promote better sampling behaviours and increase sampling efficiency by reducing noise at all intensity levels. Like mutation strategies, these importance functions are user-designable and may be tailored to scene features or transport effects. Unlike mutation strategies, they do not involve complicated functions of light path geometry nor analyses of bias, which simplifies their implementation.

The MLT pseudocode below (Algorithm 1) combines original MLT with the 'Use of Expected Values' enhancement [Vea98], using explicit importance and normalization notation [KSKAC02].

MLT requires a real function $I(x)$ over the state space (light path space) to serve as its importance function for importance sampling, and the MLT Markov chain visits each path with a tendency proportional to this importance. In classical MLT, this function $I(x)$ is the luminance (total power across the spectrum) of the radiance $f(x)$ transported along path x, but it need not be and one can integrate path radiance $f(x)$ while sampling paths x proportional to any suitable function $I(x)$.

Line 1 estimates the normalization constant b as the total importance summed using MC path tracing. In ordinary MLT using luminance-based importance, this is essentially the average pixel brightness. The importance function $I()$ serves as a probability density function by dividing it by its total sum b. Thus, the quotient of importance functions $I(y)/I(x)$ used to compute the mutation acceptance probability in Line 5 could equivalently be the quotient of the p.d.f.s $(I(y)/b)/(I(x)/b)$. We divide by the p.d.f. I/b in Lines 6 and 7 as one does in standard MC integration.

The original MLT formulation without the 'Expected Values' modification [Vea98] was based on rejection and added the full path power $f(x)$ to only one of pixel(x) or pixel(y). That original formulation avoids the need for a normalized p.d.f., and would work with any unnormalized importance function I. But MLT nevertheless needs to compute b as it is a constant of integration required to convert the relative results of MLT into absolute pixel intensities, so its use in normalizing an importance function into a probability density function comes at no additional computational expense.

As is the case in MC integration, the MLT estimator remains unbiased for any importance function, but a poor choice of importance function will increase its variance. Ordinary MLT is based on the classical importance function of integrand magnitude (path luminance) but the next section demonstrates that better MLT importance functions are available.

4. Applications

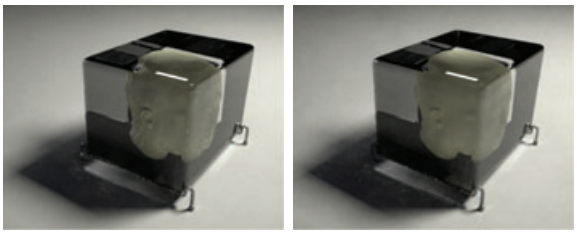
This section provides examples of importance functions other than luminance that increase estimator efficiency. These functions range from simple one-liners to sophisticated adaptive functions. They may also be chained together to create ever more elaborate sampling functions. In all our applications, our importance functions reduce to simple path throughput when applied to different paths within the same image pixel integral. In this case, an importance function which mimics the integrand is optimal. The appendix elaborates.

4.1. Importance shaders

The importance function also allows one to explicitly target parts of the light transport integral. Consider the following challenging rendering problem where a head is illuminated and viewed only through refractive transmission with the glass enclosing it. For this case, we can design an importance function that acts like a programmable surface shader in path space.

We implemented an luminance-based importance function that emphasizes paths of the form $L(S|D) * HS * E$ (where H indicates 'head') by weighting them by 10 times their luminance. This resembles user-guided path mutations [FCL05], but is much simpler to implement because there are no transitional probabilities to compute. This is shown in Figure 3.

By targeting these paths, we are able to significantly reduce the noise of this difficult to sample part of the integrand. Of course, this 'steals' samples from other areas of the image which increases the variance of hopefully less conspicuous image regions.



Luminance MLT Head-Path MLT

Figure 3: *MLT renderings using luminance importance (left panel) versus using a custom importance 'shader' (right panel) that emphasizes (by a factor of 10) paths that include the head. Custom importance shaders provide control over noise distribution, exhibiting less noise around the head, but at the expense of increased noise in the floor's caustics and soft shadows.*

4.2. Multi-stage metropolis light transport

Veach [Vea98] proposed two-stage MLT to more evenly distribute samples across the image. It was later mentioned by Pharr [Pha01] motivated by the problem of sampling motion blur. Two-stage MLT attempts to equalize the sampling rate by normalizing the scalar contribution function per pixel.

Section 11.5 of Veach's Ph.D. dissertation [Vea98] argues that the variance of pixel j is proportional to its ideal brightness, I_j, its standard error is proportional to $\sqrt{I_j}$, and its relative error is proportional to $1/\sqrt{I_j}$. Sampling proportional to relative error is desirable, because the HVS is sensitive to differences in contrast. Instead of sampling proportional to the luminance of $f(x)$, a renderer which samples proportional to $f(x)/I_j$ will promote stratification. Finally, when samples are deposited to the image, they should be weighted by an estimate of I_j to remain unbiased.

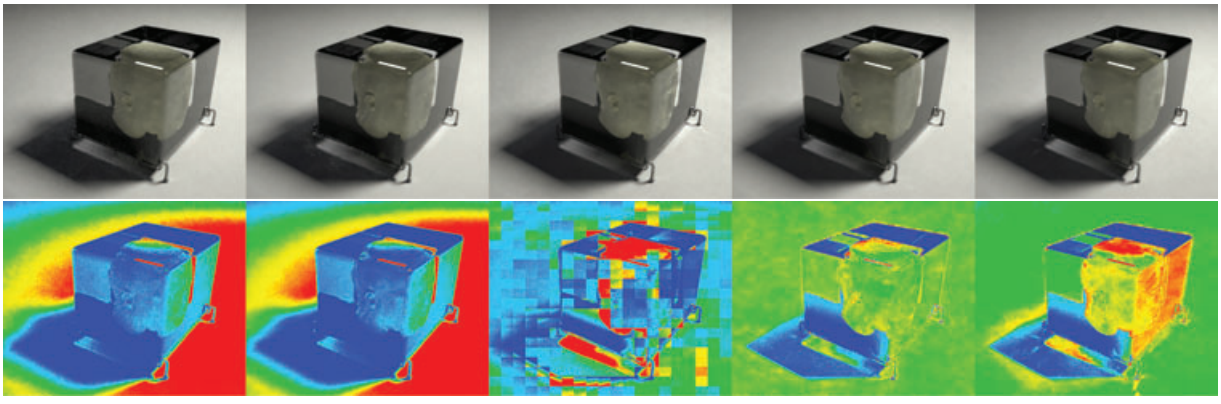
This technique is an instance of importance sampling, and we can describe it in terms of a user-defined importance function for MLT. Veach and Pharr [Vea98, Pha01] recommended a two-stage sampling process, wherein a renderer initially computes a low resolution test image I_0 with a low sampling density. The pixels of this image are used as an estimate of I_j for the second stage of the process, which samples with importance $f(x)/I_j$. Unfortunately, neither author provides sufficient detail for robust implementation. For example, it is unclear how many samples the initial estimate should consume or how the resolution of the test image should relate to the final. We found that straightforward applications of this method sometimes produced results inferior to an MLT algorithm sans this optimization.

Our experiments reveal that this method hinges critically on the quality of the test image, with poor estimates actually hindering stratification and increasing noise, as shown in Figure 4. We are also concerned that the test image's samples are discarded, especially when the test image needs to be of such high quality to be useful.

Multi-stage MLT extends two-stage MLT by bootstrapping itself iteratively with estimates of the ideal pixel distribution that get progressively more accurate over the course of the render. Because two-stage MLT is unbiased, multi-stage MLT is unbiased by induction.

We implement multi-stage MLT using the framework of user-defined importance functions. For simplicity, we assume a square power-of-two resolution image. We first create a multi-resolution importance function $I_k(x)$, where the importance function I_k divides the image into k^2 regions, for k ranging from one to the image resolution. The function $I_k(x)$ returns the total image intensity (accumulated so far) in the region of I_k pierced by path x.

Multi-stage MLT operates in two phases, as shown in Algorithm 2. The first phase doubles the size of the test image



ERPT MLT 2-Stage MLT Multistage MLT Noise-Aware MLT

Figure 4: *Renderings of a head in a glass cube above their associated sampling densities.*

Algorithm 2: Multi-stage MLT

```
for k = 1;  k ≤ n/2;  k ×= 2 do
    Let b = total current image intensity
    Exec MLT w/ kM rays and importance I_k
for k = n/2;  k ≤ n;  k += (n−n/2)/10 do
    Let b = total current image intensity
    Exec MLT w/ kM rays and importance I_k
```

until it reaches half the size of the final rendered output, consuming a quarter of the total ray budget. The second phase linearly grows the size of the test image over 10 iterations.

Following Veach and Pharr, the algorithm samples proportional to a normalized version of the image distribution to promote stratification. Multi-stage MLT takes this to its logical conclusion and progressively renders the estimate with the same algorithm. Because each of these intermediate estimates is an unbiased estimate of the ideal image distribution, they are accumulated to create the final unbiased estimate. In this way, every ray spent towards each intermediate estimate also counts towards the final estimate.

4.3. Noise-aware metropolis light transport

Our normalizing importance function encourages stratification by equalizing pixel importance and frees us to target arbitrary sampling patterns. One obvious pattern to target is a density directly proportional to image noise. Multi-stage MLT makes it possible to implement a noise-based importance function, based on a measure of the noise of the current image estimate iteratively improved by multi-stage MLT. Ashikhmin *et al.* studied MLT's variance properties in a theoretical framework [APSS01], but we are aware of no published algorithm which estimates the noise content of an MLT render in progress. This necessitates our noise estimation algorithm we detail below.

Noise-aware sampling methods are well studied in the traditional MC rendering literature, and methods exist to prioritize sampling effort which produce both biased and unbiased estimates. Some of these methods characterize noise by computing a statistical estimate of *physical* error, whereas others employ perceptual-based models of the HVS to estimate *visual* error. To our knowledge, none of these methods have been applied to MCMC rendering.

4.3.1. *Psychovisual methods*

One class of noise detection methods employs models of the HVS, and may not necessarily require knowledge of the statistical properties of some process which produced the image. Bolin and Meyer [BM98] integrated the visible differences predictor [Dal93] into an adaptive sampling algorithm. It predicts those areas of an image which the HVS is likely to perceive as noisy. They demonstrated this method in the context of estimating direct lighting, and Volevich *et al.* [VMKK00] extended it to the context of global illumination. However, a freely available implementation of this algorithm [MDMS05] revealed its parameters required extensive user tuning.

Ramasubramanian *et al.* [RPG99] also incorporated a computational model that predicts the HVS's response to visual artefacts. Specifically, their technique accounts for the HVS's loss of sensitivity in areas of high spatial frequency, for example texture, often referred to as *visual masking* [FSPG97]. This approach counterintuitively devotes *less* sampling effort to areas of high spatial frequency. The method assumes that a direct lighting solution is available to provide an initial estimate of spatial frequency, which is ill-suited for the kind of scenes that rely on MLT's ability to find obscure light paths. Furthermore, when applying this approach to multi-stage MLT, the texture detection would also detect noise and dedicate fewer samples to reduce its variance.

4.3.2. *MC methods*

Estimating the variance of a Markov chain is not as straightforward as it is for an MC process. A common way to reduce the variance of a traditional MC renderer is to explicitly stratify the work into a number of sampling domains (e.g. pixels). Then in a second pass, dedicate additional MC rendering work to the high-variance domains. But this approach is not feasible for MCMC approaches due to the irregular way the process visits the pixels.

Numerically stable incremental algorithms exist that can simultaneously maintain estimates of mean and variance [Wes79, Wel62]. We found these difficult to adapt to our MCMC renderer because they assume a uniform number of samples per domain and MCMC rendering is so difficult to stratify.

4.3.3. *MCMC methods*

We also investigated algorithms which specifically estimate the variance of Markov chains [CC96]. Of these, 'batch means' [Con63, Fis78, LC78] seems to be the most common in the MCMC field at large, and is easy to implement [AFS97]. Under certain conditions, this method produces a biased but consistent estimate of variance, by partitioning the Markov chain into a set of subchains, or batches. Each of these batches' means are computed separately from the mean of the whole chain. The sample variance of this set of means is said to be the 'batch means variance'.

In our implementation, batch means did not produce variance images that corresponded with our notion of image noise. We presume this was due to the bias inherent in the method. The batching process appears to act as a low pass

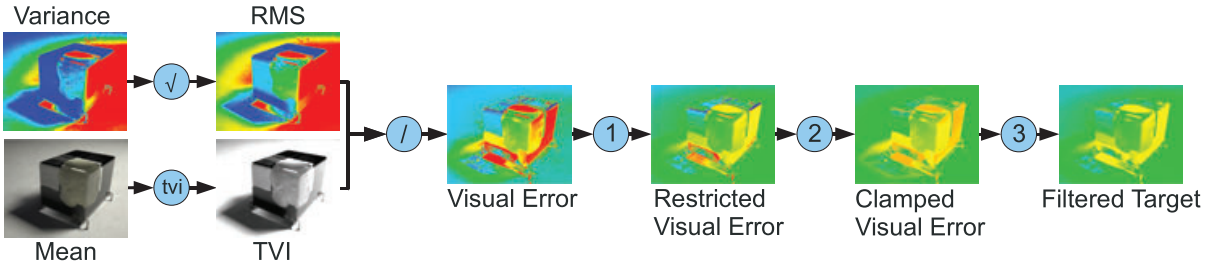


Figure 5: *Our noise-aware renderer combines an estimate of variance and mean to produce a target distribution which is proportional to perceptually significant image noise.*

filter that smooths out high variance inside pixels, which is precisely the noise we wish to target.

4.3.4. *Noise-aware MLT*

We base a new approach on the definition of the variance integral, which we integrate simultaneously with the mean. Because our variance estimate may be incrementally updated in the same way as our mean estimate, its cost is essentially free in both time and memory.

In the variance integral

$$V = \int_\Omega (f(x) - I)^2 \mathrm{d}\mu(x), \tag{2}$$

$f(x)$ is some integrand and $I = \int_\Omega f(x) d\mu(x)$ is the familiar mean integral we usually estimate with Monte Carlo methods. Because the true value of I is unknown and not analytic, we cannot compute an unbiased estimate of V by simply applying Monte Carlo to the V integral. However, we can compute a consistent estimate of V by substituting an estimate $\bar{I}$ for I

$$V \approx \frac{1}{N} \sum_1^N \frac{\left(\frac{f(x_i)}{p(x_i)} - \bar{I} \right)^2}{p(x_i)}, \tag{3}$$

where $\bar{I}$ is some unbiased estimate of I. We have implemented a noise-aware MLT on top of a multi-stage MLT, which conveniently makes such estimates available. Integrating this additional variance sum into the MLT algorithm is straightforward, as it may be updated incrementally in the same way as each pixel's mean. This approximation of variance is likely biased but consistent.

4.3.5. *A noise-aware importance function*

With this estimate of variance, we can now construct an importance function that encourages a sampling density proportional to image noise, by transforming the estimate of variance into a function that accounts for the HVS's sensitivity to contrast, and tempered to perform robustly. We multiply this target function by our previous normalized importance function $f(x)/I_j$ to both stratify and target perceptually important noise.

The flow of the process is depicted in Figure 5. We begin with our estimate of variance and take the square root to produce RMS, or standard error. This estimate is now proportional to the physical error of the estimate; however, we can increase visual efficiency if we target a measure of visual error. One way to approximate visual error is to divide RMS by the *threshold-versus-intensity* function as applied to the mean image [Fan06]. We measure *tvi* using the current mean image, but downsample it before the division. We use Ward *et al.*'s piecewise approximation to the tvi function [WRP97].

The final three steps involve adjusting the target image to ensure a base level of numerical robustness and so that it fulfils the unbiasedness restrictions of an importance function. Step 1 restricts the contrast of the visual error image so that the ratio of the 95th percentile to the 5th percentile is no greater than 2.

Step 2 removes extreme highs and lows in the target to avoid grossly unfair sampling, by clamping the range of pixel values to the values of the 5th and 95th percentiles, which also removes any zeros from the target distribution. Step 2 also locates yet unsampled parts of the domain, identified by black pixels in the mean image. We set these pixels in the target distribution to the 95th percentile value to encourage their sampling, which is especially crucial early in the sampling process.

Step 3 removes high spatial frequencies from the target using a bilateral filter [TM98].

5. Results

5.1. Implementation details

Our MLT implementation is based on the mutation rule described by Kelemen *et al.* [KSKAC02]. We mutate hyperpoints in the unit hypercube which then get mapped to light paths. This scheme has the desirable property that the transitional probability density ratio $\frac{T(y \to x)}{T(x \to y)}$ is always unity

because these mutations are symmetric. This mutation rule has a single user parameter, which is the probability of proposing a 'large step' drawn uniformly at random. We set this parameter to 0.001 for all scenes except the easily sampled Cornell box, where it was set to 0.5. Like Kelemen's original algorithm, our implementation combines stratified large steps with multiple importance sampling, but the effect of this scheme is not significant in most of our test scenes due to the small probability of proposing a large step. This scheme provides a stratifying function similar to the lens subpath mutations of Veach's original implementation. Finally, we adopt Veach's use of expected values when depositing the result of each mutation. We use these same basic techniques in all our experiments, which we believe enables the fairest possible comparison.

Paths were generated with forward path tracing [Kaj86], except for the dark room scene, where it was necessary to use bidirectional path tracing [LW93, VG94]. The radiance of these paths was also weighted with the power heuristic [VG95]. Paths are randomly terminated with Russian roulette, which is tested after each path vertex [KA91]. In all cases, path visibility is tested with a single shadow ray. For two-stage MLT, we generated the coarse resolution estimate image using 10^6 rays.

Our ERPT implementation is based on the multi-deposition flow rule and uses sample chains of length 100 as recommended by Cline *et al.* [CTE05]. For our examples, we used one mutation per Monte Carlo sample. Experimenting with this parameter probably would have changed the quality of our results, but we expect that anything significantly different would have behaved similarly to either regular MLT or MC path tracing. In addition, we did not implement any noise reduction filter for ERPT.

Unlike Veach's original proposal, we do not perform any special separation of the estimation of direct and indirect light transport. In all our experiments, a single unified algorithm estimates all transport effects. Though this might improve the efficiency of all algorithms under comparison in the simple Cornell Box scene, all other scenes are dominated by indirect light: the lab scene's light sources are encased behind glass surfaces, the head in the glass box is illuminated entirely by caustics, and the dark room scene is illuminated entirely by indirect light that must bounce through three rooms before reaching the camera. Indeed, in these experiments, using separate algorithms would only decrease efficiency.

5.2. Methodology

To compare our results to previous work, we experimented with four different sampling regimes which increase in complexity. We measured the difficulty of sampling these scenes by the failure rate for a path's visibility test. We call this quantity *shadow rate*, and it increases as visibility grows in complexity.

To assess how well each algorithm stratifies over the image plane, we measure three statistics of the sample distribution. These are variance, skew, and kurtosis. Skew measures a distribution's asymmetry. For a population of N samples with sample mean $\bar{x}$, the population skew is

$$\text{skew} = \frac{m_3}{(m_2)^{\frac{3}{2}}} = \frac{\sqrt{N}\sum_{i=1}^{N}(x_i - \bar{x})^3}{\left(\sum_{i=1}^{N}(x_i - \bar{x})^2\right)^{\frac{3}{2}}}, \tag{4}$$

where m_3 is the third central moment and m_2 is sample variance. A majority of pixels in a positively skewed sampling distribution receive more samples than the mean, whereas a negatively skewed one would receive less than the mean. We seek to minimize the magnitude of skew.

Kurtosis measures the prominence of peaks in a distribution. It is defined

$$\text{kurtosis} = \frac{m_4}{(m_2)^2} = \frac{\sqrt{N}\sum_{i=1}^{N}(x_i - \bar{x})^4}{\left(\sum_{i=1}^{N}(x_i - \bar{x})^2\right)^2} - 3, \tag{5}$$

where m_4 is the fourth central moment. Large kurtosis indicates a sampling distribution with much variance due to infrequent large deviations from the mean. We seek to minimize kurtosis.

The sampling density was computed by rendering a separate image of the weights a and $(1 - a)$ associated with each sample.

Our efficiency and error metrics follow the methodology of Lai *et al.* [LFCD07]. To assess the efficiency of each algorithm, we first render a reference image by running MLT for a long time. The error of a test image is measured by the sum

$$\text{error} = \frac{1}{N_{\text{pixels}}}\sum_{\text{pixels}}(L(\text{ref}) - L(\text{test}))^2, \tag{6}$$

where L maps a pixel's spectral radiance to scalar luminance.

Visual efficiency is computed by taking into account the HVS's sensitivity to constrast with the threshold-versus-intensity function *tvi*. The visual error of a test image is measured using the sum

$$\text{visual error} = \frac{1}{N_{\text{pixels}}}\sum_{\text{pixels}}\frac{|L(\text{ref}) - L(\text{test})|}{tvi(\text{ref})}. \tag{7}$$

We measure efficiency by accounting for both error and the number of rays spent, R as $1/(R \times \text{error})$. We likewise define visual efficiency as $1/(R \times \text{visual error})$.

Table 1: *Statistics of our test scenes: number of rays cast, percentage of paths whose shadow rays were found to be blocked (Shdw), mutation acceptance rate (Acc.), variance (Var.), skew and kurtosis (Kur.) of sample distribution, error, efficiency (Eff.), visual error (VErr.), and visual efficiency (VEff.).*

Scene.	Alg.	Rays	Shdw	Acc.	Var.	Skew	Kur.	Error	Eff.	VErr.	VEff.
Cornell	Path	123M	10.3%	N/A	0.0000	0.000	−3.000	0.168	48.3e-9	1.93	4.20e-9
Box	MLT	130M	3.89%	49.0%	3.7320	8.217	67.377	0.169	45.5e-9	1.67	4.62e-9
	ERPT	130M	3.94%	84.7%	3.7322	8.249	68.127	0.172	44.7e-9	1.88	4.10e-9
	2MLT	131M	4.57%	44.3%	0.1558	1.618	18.235	0.170	44.8e-9	1.58	4.84e-9
	MMLT	131M	4.55%	44.2%	0.0936	−0.794	−0.666	0.182	44.2e-9	1.54	4.96e-9
	NAMLT	131M	4.60%	43.5%	0.1035	−0.464	−0.613	0.181	42.2e-9	1.56	4.88e-9
Head	Path	554M	27.6%	N/A	0.0000	0.000	−3.000	2.55e-3	707e-9	1.33	1.36e-9
	MLT	500M	2.31%	90.1%	0.1719	0.712	0.157	1.76e-3	1140e-9	1.01	1.98e-9
	ERPT	520M	12.1%	90.0%	0.1717	0.790	0.823	2.00e-3	959e-9	1.14	1.69e-9
	2MLT	501M	7.01%	68.1%	1.2357	11.457	241.915	2.77e-3	720e-9	1.08	1.84e-9
	MMLT	501M	4.11%	82.5%	0.0346	−1.368	2.921	1.78e-3	1120e-9	0.91	2.20e-9
	NAMLT	501M	4.63%	80.1%	0.0486	6.663	528.310	1.89e-3	1050e-9	0.91	2.18e-9
Lab	Path	59M	74.3%	N/A	0.0000	0.000	−3	4647	3.65e-12	8.80	1.93e-9
	MLT	50M	14.6%	58.5%	1425.1	567.4	362057	841	23.8e-12	7.46	2.68e-9
	ERPT	48M	31.5%	63.1%	1864.7	587.7	383456	1170	17.8e-12	7.45	2.79e-9
	2MLT	51M	67.1%	12.1%	90.576	42.63	2755	2210	8.87e-12	9.14	2.15e-9
	MMLT	50M	17.4%	41.8%	34.414	334.2	142746	2303	8.68e-12	5.15	3.88e-9
	NAMLT	50M	16.2%	48.8%	14.695	509.7	299061	5757	3.47e-12	4.19	4.78e-9
Dark	Path	52M	97.4%	N/A	0.0000	0.000	−3.0	1.013	19.0e-9	16.07	1.20e-9
Room	MLT	50M	29.6%	54.6%	1.0260	4.623	23.5	0.045	448e-9	5.43	3.68e-9
	ERPT	65M	48.7%	54.5%	1.1449	4.859	28.8	0.049	315e-9	5.08	3.03e-9
	2MLT	51M	31.2%	49.9%	0.4051	4.053	52.3	1.171	16.7e-9	6.94	2.83e-9
	MMLT	51M	30.4%	47.1%	0.0693	3.021	46.6	0.458	42.8e-9	4.70	4.17e-9
	NAMLT	51M	31.6%	45.2%	0.0952	2.626	29.2	0.324	60.6e-9	4.59	4.27e-9

Note: Results shown for path tracing, MLT, ERPT, two-stage MLT, multi-stage MLT and noise-aware MLT.

Because we have not prioritized computational efficiency in our implementation of MLT, we decline to report statistics on clock time. The images computed in our results took from 5 to 30 min on a modern workstation. The addition of our multi-stage techniques did not noticeably increase render time over renders without.

5.3. Analysis

We compare the results of our generally applicable Multi-stage and noise-aware methods to previous work and summarize them in Table 1. We first observe that even though ERPT is designed to encourage stratification, its samples do not stratify any better than MLT's. This is evidenced by their nearly identical sampling distribution statistics across all regimes. Figure 6 makes this apparent. Even in the Cornell box, a very easy scene to sample, ERPT's skew and kurtosis is no less than MLT's. This is not surprising, as ERPT and MLT share the same importance function that causes their samples to gravitate towards brighter image areas. As visibility becomes more complex, ERPT's reliance on frequently shadowed Monte Carlo samples becomes a severe weakness and stratification worsens relative to MLT's.

Two-stage sampling occasionally offers some improvement in stratification, but the performance gain with multi-stage sampling, essentially a robust version of the two-stage technique, is always significant. In general, the extent to which we can successfully stratify depends on the complexity of visibility. For example, in the Cornell box, unshadowed paths are readily available and stratification is magnitudes greater than with MLT or ERPT. By contrast, in the head in glass scene of Figure 4, multi-stage MLT undersamples the shadowed region because light paths here are rarer than those elsewhere. Notably, noise-aware sampling improves stratification over MLT and ERPT, even though it targets an uneven sampling distribution.

Stratification is not an end in itself, and we are most interested in minimizing visual error. Over all cases, our importance sampling techniques increase visual efficiency over all previous approaches and as expected decrease absolute efficiency. In the Cornell box, there is little improvement to be gained even after nearly perfect stratification. Nevertheless, noise-aware MLT directs sampling effort towards the indirectly lit ceiling and glossy caustic, both of which become noticeably smoother.

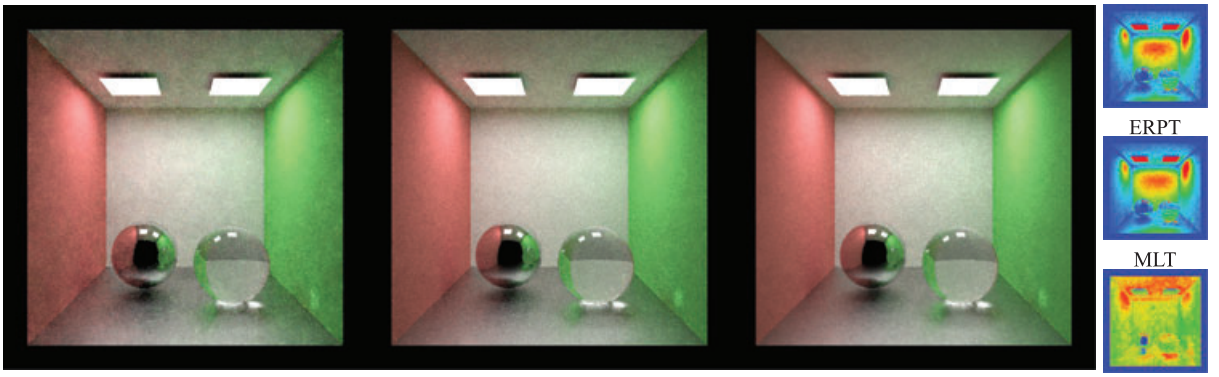


Figure 6: *ERPT, MLT and noise-aware MLT in the Cornell box scene and the corresponding sampling densities.*

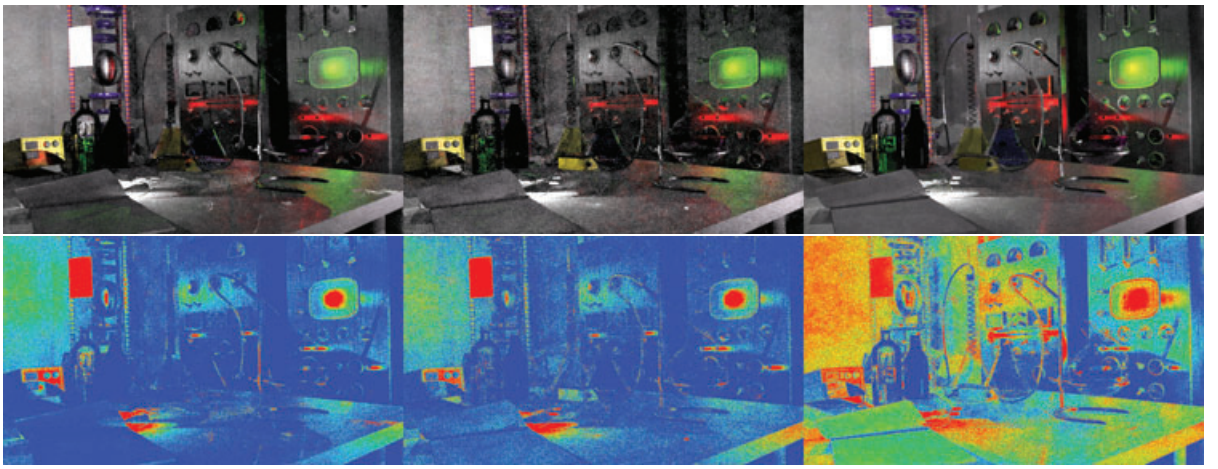
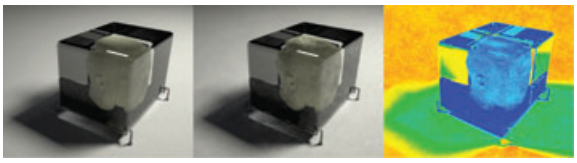


Figure 7: *This scene contains many difficult to sample light transport paths: glossy brushed-metal reflection, glass object caustics and indirect lighting from many light sources. Our investigation of importance sampling for Metropolis light transport has led to a multi-stage version that supports using a measure of image noise to adaptively increase the sampling rate, yielding a noise-aware Metropolis light transport that stratifies Markov chain path samples across the image plane better than previous approaches. All three renderings used approximately the same number of rays. The sampling density is shown below the rendered output.*

The head in glass scene benefits well from our ability to detect high noise in the caustic and penumbral regions. Our heuristic also correctly detects the head as noisier than the background and increases sampling there. In addition, the noise-aware technique has discovered the faint caustics cast into the shadow from the metal stand which are not present in any other version of the image.

The lab scene is primarily a challenging direct lighting case as it is lit by 37-point light sources. Most of MLT's and ERPT's effort is spent sampling the bright surfaces closest to these. As a result, regions such as the metal sphere and computer screen are quite smooth. The rest of the scene, which includes glossy brushed-metal surfaces are significantly noisier. This scene is also a challenge for our multi-stage sampler to stratify because of the high shadow rate. This is evidenced by the higher sampling density variance over the two easier scenes. However, Figure 7 shows our noise-aware technique does well to spread out the sampling density over the metal surfaces and dimly lit background.

The final dark room scene is a difficult indirect lighting situation where light must bounce through three rooms before reaching the camera. Light paths reaching the wall on the right side of the image are doubly difficult to find because



MC Path Tracing MLT w/$I(x) = C$ Sampling density

Figure 8: *$I(x) = C$ performs poorly.*

they are both the dimmest and the longest. Nevertheless, these paths have been correctly identified as visually important, and noise-aware MLT significantly smooths the large, dim portions of the image. This improvement has been paid for with the increase in noise in the bright part of the left wall.

Interestingly, the mutation proposal acceptance rate seems to have little effect on efficiency, which contradicts the prevailing wisdom that high acceptance rates should be a goal.

6. Conclusions

We have presented a novel application of importance sampling to Markov chain processes and demonstrated how programmable importance functions can significantly improve the visual efficiency of an MCMC rendering algorithm. These functions allow direct user control over a sampling distribution, or may automatically adapt to target areas of high image noise. Our techniques are naturally unbiased, which makes them straightforward to implement.

We see many avenues for future work. One is to apply multiple importance sampling [VG95] to MLT with a collection of different importance functions that could vary over the course of a rendering process. Our noise detection heuristic does not account for tone mapping, the inclusion of which could further reduce visual error by incorporating an estimate of the final displayed image. Stratification over domains other than the image plane could be possible, including the surfaces of light sources, or in the case of particle tracing, the scene surface itself. Finally, because our actual sampling density does not always meet our target goal, it may be possible to incorporate a feedback mechanism to compensate for discrepancies.

Appendix

$I(x) = constant$ might seem like it would naturally encourage stratification. In fact, as Figure 8 shows, it does not. Because $\frac{I(y)}{I(x)} = 1$, every proposal is accepted and the stratification success becomes totally dependent on the mutation proposal density. Unfortunately, we have no guarantee of stratification for the proposal density in general. Visual efficiency worsens, becoming comparable to Monte Carlo path tracing, because we have effectively turned off importance sampling within each pixel domain. For this reason, successful importance functions like the ones we have described should reduce to $\frac{I(y)}{I(x)} = \frac{f(y)}{f(x)}$ when paths x and y project to the same pixel.

Acknowledgmeents

The authors thank NVIDIA, Keenan Crane and David Cline for providing the models used in the results.

References

[AFS97] Alexopoulos C., Fishman G. S., Seila A. F.: Computational experience with the batch means method. In *Proceedings of Winter Simulation Conference* (1997), pp. 194–201.

[APSS01] Ashikhmin M., Premoze S., Shirley P., Smits B.: A variance analysis of the metropolis light transport algorithm. *Computers and Graphics 25*, 2 (2001), 287–294.

[Arv01] Arvo J.: Stratified sampling of 2-manifolds. In *State of the Art in Monte Carlo Ray Tracing for Realistic Image Synthesis*. SIGGRAPH Course Notes, 2001, pp. 41–64.

[BM98] Bolin M. R., Meyer G. W.: A perceptually based adaptive sampling algorithm. In *Proceedings of SIGGRAPH* (1998), pp. 299–309.

[CC96] Cowles M. K., Carlin B. P.: Markov chain Monte Carlo convergence diagnostics: A comparative review. *Journal of the American Statistical Association 91*, 434 (1996), 883–904.

[Con63] Conway R. W.: Some tactical problems in digital simulation. *Management Science*, 10 (1963), 47–61.

[CTE05] Cline D., Talbot J., Egbert P.: Energy redistribution path tracing. *ACM TOG 24 (Proc. SIGGRAPH)*, 3 (2005), 1186–1195.

[Dal93] Daly S.: *The Visible Differences Predictor: An Algorithm for the Assessment of Image Fidelity*. MIT Press, Cambridge, MA, 1993, pp. 179–206.

[Fan06] Fan S.: *Sequential Monte Carlo Methods for Physically Based Rendering*. PhD thesis, 2006.

[FCL05] Fan S., Chenney S., Lai Y.: Metropolis photon sampling with optional user guidance. *Eurographics Symposium Rendering* (2005), 127–138.

[Fis78] Fishman G.: Grouping observations in digital simulation. *Management Science 24*, 5 (1978), 510–521.

[FPSG96] Ferwerda J. A., Pattanaik S. N., Shirley P., Greenberg D. P.: A model of visual adaptation for

realistic image synthesis. In *Proceedings of SIGGRAPH* (1996), pp. 249–258.

[FSPG97] Ferwerda J. A., Shirley P., Pattanaik S. N., Greenberg D. P.: A model of visual masking for computer graphics. In *Proceedings of SIGGRAPH* (1997), pp. 143–152.

[JC95] Jensen H. W., Christensen N. J.: Photon maps in bidirectional Monte Carlo ray tracing of complex objects. *Computer Graphics 19*, 2 (1995), 215–224.

[KA91] Kirk D., Arvo J.: Unbiased sampling techniques for image synthesis. *Computer Graphics (Proc. SIGGRAPH) 25*, 4 (1991), 153–156.

[Kaj86] Kajiya J. T.: The rendering equation. In *Proceedings of SIGGRAPH* (1986), pp. 143–150.

[KSKAC02] Kelemen C., Szirmay-kalos L., Antal G., Csonka F.: Global illumination: A simple and robust mutation strategy for the metropolis light transport algorithm. *CGF 21*, 3 (2002), 531–540.

[LC78] Law A. M., Carson J. S.: A sequential procedure for determining the length of a steady-state simulation. *Operations Research 27*, 5 (1978), 1011–1025.

[LFCD07] Lai Y.-C., Fan S., Chenney S., Dyer C. R.: Photorealistic image rendering with population Monte Carlo energy redistribution. In *Proceedings of Eurographics Symposium Rendering* (2007), pp. 287–296.

[LRR04] Lawrence J., Rusinkiewicz S., Ramamoorthi R.: Efficient brdf importance sampling using a factored representation. *ACM Transactions Graphics (Proc. SIGGRAPH) 23*, 3 (2004), 496–505.

[LW93] Lafortune E., Willems Y.: Bi-directional path tracing. In *Proceedings of Computer Graphics* (1993), pp. 95–104.

[MDMS05] Mantiuk R., Daly S., Myszkowski K., Seidel H.-P.: Predicting visible differences in high dynamic range images - model and its calibration. In *Human Vision and Electronic Imaging X, Proceedings of IS&T/SPIE Symposium on Electronic Imaging* (2005), vol. 5666, pp. 204–214.

[PH04] Pharr M., Humphreys G.: *Physically Based Rendering*. Morgan Kaufmann, San Francisco, 2004.

[Pha01] Pharr M.: Metropolis sampling. In *State of the Art in Monte Carlo Ray Tracing for Realistic Image Synthesis*. SIGGRAPH Course Notes, 2001, pp. 91–118.

[RPG99] Ramasubramanian M., Pattanaik S. N., Greenberg D. P.: A perceptually based physical error metric for realistic image synthesis. In *Proceedings of SIGGRAPH* (1999), pp. 73–82.

[SIP07] Segovia B., Iehl J., Peroché B.: Metropolis instant radiosity. *Proceedings of Eurographics Symposium Rendering* (2007).

[SWZ96] Shirley P., Wang C., Zimmerman K.: Monte carlo techniques for direct lighting calculations. *ACM Transactions on Graphics 15*, 1 (1996), 1–36.

[TM98] Tomasi C., Manduchi R.: Bilateral filtering for gray and color images. In *Proceedings of ICCV '98* (1998), p. 839.

[Vea98] Veach E.: *Robust Monte Carlo Methods for Light transport Simulation*. PhD thesis, Stanford University, 1998.

[VG94] Veach E., Guibas L.: Bidirectional estimators for light transport. *Proceedings of Eurographics Workshop on Rendering* (1994), pp. 147–162.

[VG95] Veach E., Guibas L.: Optimally combining sampling techniques for Monte Carlo rendering. *Proceedings of SIGGRAPH* (1995), pp. 419–428.

[VG97] Veach E., Guibas L. J.: Metropolis light transport. In *Proceedings of SIGGRAPH* (1997), pp. 65–76.

[VMKK00] Volevich V., Myszkowski K., Khodulev A., Kopylov E. A.: Using the visual differences predictor to improve performance of progressive global illumination computation. *ACM Transactions on Graphics 19*, 2 (2000), 122–161.

[Wel62] Welford B. P.: Note on a method for calculating corrected sums of squares and products. *Technometrics 4*, 3 (1962), 419–420.

[Wes79] West D. H. D.: Updating mean and variance estimates: An improved method. *CACM 22*, 9 (1979), 532–535.

[WRP97] Ward G., Rushmeier H., Piatko C.: A visibility matching tone reproduction operator for high dynamic range scenes. *IEEE TVCG 3*, 4 (1997), 291–306.