Eurographics Symposium on Rendering (2025)
B. Wang and A. Wilkie (Editors)

Convergence Estimation of Markov-Chain Monte Carlo Rendering

Rui Yu[1,2], Guangzhong Sun[1], Shuang Zhao[3], Yue Dong[2]

[1]University of Science and Technology of China, [2]Microsoft Research Asia, [3]University of California, Irvine

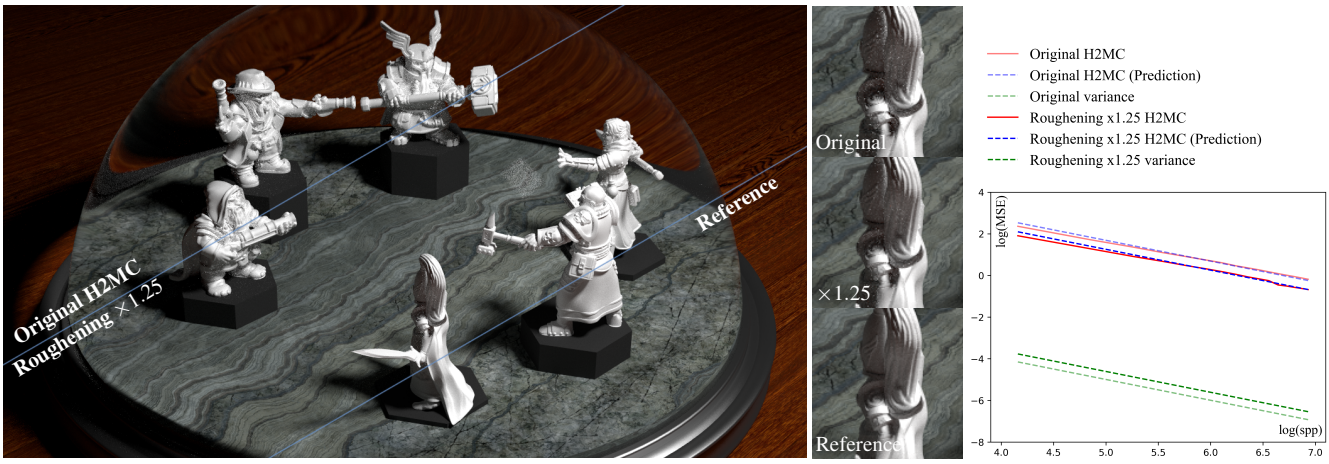

Figure 1: *Rendering results using the original H2MC and with a roughening scheme. The roughening scheme modifies the target distribution of H2MC by roughening the specular BRDF by a factor of* $\times 1.25$*, yet achieves unbiased rendering to the same results as the original scene. Our theory can estimate the expected Mean Squared Error (*MSE*) of the rendering with increasing samples per pixel (spp), as shown by the blue dashed lines, which match with the actual* MSE *depicted by the solid curves. To obtain a reliable measure of the actual* MSE*, the MCMC algorithm is run several times to calculate the average* MSE*.*

Abstract

We present a theoretical framework for estimating the convergence of Markov-Chain Monte Carlo (MCMC) rendering algorithms. Our theory considers both the variance and the correlation between samples, allowing for quantitative analyses of the convergence properties of MCMC estimators. With our theoretical framework, we devise a Monte Carlo (MC) algorithm capable of accurately estimating the expected MSE *of an MCMC rendering algorithm. By adopting an efficient rejection sampling scheme, our MC-based* MSE *estimator yields a lower standard deviation compared to directly measuring the* MSE *by running the MCMC rendering algorithm multiple times. Moreover, we demonstrate that modifying the target distribution of the Markov chain by roughening the specular BRDF might lead to faster convergence on some scenarios. This finding suggests that our estimator can serve as a potential guide for selecting the target distribution.*

CCS Concepts

• ***Computing methodologies*** → ***Ray tracing;***

1. Introduction

Markov Chain Monte Carlo (MCMC) is a powerful tool for drawing samples from complex distributions. In computer graphics, MCMC is widely used in physics-based rendering. Since its initial introduction by [VG97], most research has focused on developing improved sample mutation schemes to sample critical light paths more effectively. Specific mutation schemes have been introduced to enhance sampling over challenging specular paths [CA00, JM12, KHD14, HKD15], complex geometries [OHHD18], leveraging higher order derivatives [LLR*15], momentum and caches [LZBG20]. The combination of these mutation schemes has also been extensively studied, with adaptively multiplexing [HKD14] or combining path space mutation and primary space [KSKAC02] mutations [OKH*17, Pan17, BJNJ17, BJ19]. In practice, MCMC rendering algorithms can sample "difficult" light paths efficiently by: (i) starting with a potentially high-contribution "seed" path;

DOI: 10.2312/sr.20251179

and (ii) iteratively mutating this path to explore correlated "nearby" paths.

To analyze the convergence of MCMC rendering algorithms, early studies [APSS01] indicate that the convergence rate of MCMC rendering is $\Theta(1/N)$, even with correlated samples. Further, these correlations can negatively impact the convergence speed of MCMC algorithms [KSKAC02] and lead to non-uniform convergence rates. Although many empirical studies have explored how sample correlation affects the efficiency of MCMC rendering, theoretical analysis and efficient measurement of the convergence of MCMC rendering algorithms have remained largely unexplored in computer graphics.

In this paper, we present a theoretical analysis demonstrating that the convergence speed of an MCMC rendering algorithm is determined by two factors: the variance of the steady-state distribution and the correlation between the samples. We also design a Monte Carlo method to estimate both the variance and the correlation effects of a given MCMC rendering process, allowing efficient computation of its expected MSE (at a given sampling rate). Additionally, we develop a rejection sampling scheme capable of significantly improving the performance of our estimator.

We validate our theory by predicting the convergence of both PSSMLT [KSKAC02] and H2MC [LLR*15] algorithms on three typical scenes. With an equal computation budget, our MSE estimator yields lower standard deviation compared to directly measuring the MSE by running the MCMC rendering algorithm multiple times. Figure 1 illustrates a plot of the predicted MSE changing with increasing samples, compared to the actual measured MSE of the MCMC rendering. The actual MSE of the MCMC rendering is averaged from multiple independent MCMC renderings to eliminate randomness. The prediction of MSE aligns closely with the actual measurements.

Concretely, our contributions include:

- We present a theoretical framework for estimating the convergence of MCMC rendering algorithms, considering both the variance and the correlation between samples.
- We devise a Monte Carlo algorithm with rejection sampling, capable of accurately and efficiently estimating the MSE of an MCMC algorithm at a given sampling rate.
- We demonstrate that using a Markov chain's stationary distribution proportional to the measurement contribution of light paths does not always yield the fastest convergence. For example, on scenes with complex light-transport effects, a well-chosen roughening scheme can enhance convergence speed.

2. Related Works

MCMC rendering. Markov Chain Monte Carlo (MCMC) refers to a class of algorithms that generate samples from a target distribution by constructing a Markov chain that has the target distribution as its steady-state distribution [GRS95, RCC99]. The Metropolis-Hastings algorithm [MRR*53, Has70] is a widely used MCMC algorithm, which uses a proposal distribution and an acceptance probability to create a Markov chain that converges to the target distribution.

In physically based rendering, MCMC algorithms are employed to generate light paths that contribute to the final image [ŠK20]. The Metropolis Light Transport (MLT) algorithm [VG97] is an MCMC algorithm that generates light paths by modifying existing paths, which operates in path space. To simplify the algorithm, Kelemen *et al.* [KSKAC02] proposed MCMC algorithms that operate in primary sample space, making the proposal independent of specific rendering effects.

Various path proposal strategies have been introduced to explore challenging light paths. The first derivatives of the half-vector of a light path are used to guide the path proposal to sample difficult specular paths [CA00, JM12, KHD14, HKD15]. Li *et al.* [LLR*15] leveraged both first and second order derivatives for a Hamiltonian Monte-Carlo inspired path proposal scheme. Langevin Monte Carlo [LZBG20] incorporates adaptive preconditioning and momentum schemes that achieve efficient mutation with just first-order derivatives. The image domain spatial gradient can also be used to guide the MCMC sampling [LKL*13]. Geometry-aware MLT [OHHD18] improves sampling over complex visibility cases by adapting step sizes according to scene geometry. By efficiently sampling ensembles of transport paths, Ensemble MLT [BRSMD21] introduced a series of transition kernels that eliminate the need for world space caching.

Different sampling schemes can be adaptively multiplexed [HKD14] to enhance performance. Additionally, mutations in the path space and primary sample space can be combined to leverage the advantages of both spaces [OKH*17, Pan17, BJNJ17]. Finding a proper MLT algorithm for a specific scene has also been investigated [BJ19]. All those MCMC rendering algorithms aim to improve mutation strategies for more efficient light transport exploration. Our theory is developed to analyze and estimate their convergence properties. While most algorithms use path energy contribution as the target distribution, our theory suggests that the target distribution is also a new avenue for future explorations.

Customize stationary distribution. In the original MLT algorithm, Veach [Vea98] discussed the potential choices for the target distribution function and proposed a two-stage MLT algorithm to equalize the sampling rate over pixels by normalizing the target distribution, thus reducing relative error. Hoberock and Hart [HH10] extended this concept into a multi-stage MLT, rendering the image in multiple recursive stages. Each stage employs a refined normalization based on the accumulated samples from previous stages. Recently, Zirr *et al.* [ZD20] introduced an analytical approach to stratification and adaptive sampling in MCMC, enabling adaptive sampling that solely uses forward path construction.

Markov chains with different customized target distributions are also widely used in replica exchange MCMC algorithms [SW86], also known as parallel tempering, and were introduced to MCMC rendering by Kitaoka *et al.* [KKK09] The parallel tempering framework allows for the use of multiple Markov chains with different target distributions, exploring the sample space more efficiently. Different levels of relaxation constants are used by Otsu *et al.* [OYH*13] to create multiple Markov chains; path regularization schemes like roughening the specular BRDFs [KD13] can also be used to create higher temperature chains [ŠK16]. Šik *et al.* [ŠOHK16] also use two Markov chains, one with a regular path

contribution target function and another with a path visibility target function in their MCMC sampler. Replica exchange can also be used in the image space [GWH20] while keeping the target distribution unchanged.

Existing works that use customized target distributions either focused on optimizing for a different goal like reducing relative error [HH10, ZD20] or combined with the replica exchange scheme [ŠOHK16, ŠK16] aiming to provide better exploration performance among multiple Markov chains with different steady-state distributions. Alternating target distribution can also be utilized for effective photon tracing [HJ11]. In our application section, we show that even for the vanilla single-chain MCMC scenario, customizing the target distribution can lead to a better convergence rate measured by MSE. Additionally, our analysis and convergence estimation can be used to predict the performance of the choice of target distributions, which could serve as a better tool for designing target distributions in the future.

Convergence analysis for MCMC rendering. In statistical studies, theoretical analysis of MCMC algorithms shows that their convergence rate is determined by sample correlation properties [Ber04]. A majority of works focus on estimating the Effective Sample Size (ESS) to assess the convergence rate of sequential Monte Carlo (SMC) algorithms [DDFG*01]. The ESS is defined as the number of independent samples that would provide the same variance as the correlated samples. The ESS can be used to estimate the convergence rate of MCMC algorithms by comparing the variance of the samples to the variance of independent samples. However, how to effectively estimate or approximate the ESS for a given MCMC algorithm is still an active research topic in statistics [MEL17, VGS*21, FCS22].

In computer graphics, early studies [APSS01] provided loose bounds on sample covariance, showing that the convergence rate of MCMC rendering is $\Theta(1/N)$ even in the presence of sample correlation. However, it has been observed that correlation significantly impacts the convergence speed of MCMC rendering algorithms [KSKAC02]. Unlike previous works offering either empirical analysis or loose bounds, our theory accurately estimates how sample correlation affects the convergence rate and successfully predicts the convergence rates of existing algorithms.

3. Preliminaries

In this section, we will first revisit some preliminaries in Markov Chain Monte Carlo. Table 1 summarizes the symbols and notations commonly used in this paper.

Physically based rendering with Markov Chain Monte Carlo (MCMC) can be formulated as computing a path integral of the measurement contribution function $f(x)$ over discretized pixels j:

$$I_j = \int_\Omega h_j(x) f(x)\,\mathrm{d}x, \tag{1}$$

where Ω is the full path space, and the image reconstruction kernel $h_j(x)$ determines which path contributes to the corresponding pixel j.

Given a target distribution $g(x)$, also defined on the path space Ω,

Notation	Description
Ω	The state space of a Markov process
$x, f(x), g(x)$	State $x \in \Omega$, its contribution, distribution function
$h_j(x), f_j(x)$	Reconstruction filter of pixel j, $f_j(x) := h_j(x)f(x)$
I_j	Reference value of pixel j. $I_j = \sum_{x\in\Omega} f_j(x)dx$
$w_j(x)$	Splat value $w_j(x) = \frac{f_j(x)}{g(x)}$.
$\lvert w_j\rangle, \lvert g\rangle$	Vector consists of $w_j(x), g(x)$ for all $x \in \Omega$
M	Transition matrix
$M^k(y,x)$	k-step transition probability ($x \to y$)
G	The limit matrix of M^k. $G := [\lvert g\rangle, \lvert g\rangle, ..., \lvert g\rangle]$
$\hat{I}_j$	Estimate of pixel j. $\hat{I}_j = \frac{1}{n}\sum_{i=1}^{n} w_j(x_i)$

Table 1: *Important notations used throughout the derivation*

we can design an MCMC process that produces a series of samples $x_1, ..., x_n$ where the final distribution follows the target distribution $g(x)$. With this, we can define a *Splat Weight Function*:

$$w_j(x) = \frac{h_j(x)f(x)}{g(x)}, \tag{2}$$

such that the image integral in Eq.(1) can be computed as the estimate of the weights of all the samples:

$$I_j = \int_\Omega h_j(x)f(x)\,\mathrm{d}x = \int_\Omega w_j(x)g(x)\,\mathrm{d}x = \lim_{n\to+\infty} \mathrm{E}(w_j(x_n)), \tag{3}$$

where the estimate of the integral is asymptotically unbiased and converges when the number of samples reaches infinity. Note that the target distribution $g(x)$ does not necessarily have to be the same as $f(x)$; any $g(x)$ that meets the condition $f(x) \neq 0 \to g(x) > 0$ is suitable, as long as a proper splat weight function $w_j(x)$ is applied, the MCMC estimator $w_j(x_i)$ is still unbiased.

In primary-sample space (PSS) MCMC, the state space of the Markov process is a hypercube containing vectors of random numbers that are mapped to light paths via a predetermined path sampling procedure. The only difference is the definition of the measurement contribution function, where for the path space samples x, $f(x)$ is the radiance contribution of the path, while for the primary space samples x, $f(x)$ is the radiance divided by the Jacobian of x that measures the projection differences between the primary space and the path space, and the PSS-MCMC rendering can be written as:

$$I_j = \int h_j(x(u))f(x(u))J(u)\,\mathrm{d}u, \tag{4}$$

where $x(u)$ maps from primary-sample space u to a path, and $J(u)$ is the Jacobian factor. Notably, $J(u)$ is the reciprocal of the probability density function pdf(x), i.e., $J(u) = \frac{1}{\mathrm{pdf(x(u))}}$, reflecting the change of variables from path space to primary-sample space. For simplification, the following derivation will be based on a Markov process defined in the primary-sample space, but the same theory can be applied to the path space as well. In later derivations, we will use x and $f(x)$ to denote a sample and its measurement contribution function, assuming all terms like the Jacobian are already considered.

4. Theoretical Analysis

In this section, we will begin with a theoretical analysis of the convergence of MCMC (Sec. 4.1), followed by the development of a Monte Carlo estimator for calculating the expected mean squared error (MSE) of an MCMC rendering algorithm (Sec. 4.2).

4.1. Variance of MCMC based integration

As expressed in Eq. (3), a valid MCMC estimator is asymptotically unbiased and converges to the ground truth for any valid $g(x)$ when the length of the Markov chain reaches infinity. A $g(x)$ is valid when it maps all the non-zero region of $f(x)$ into a positive value, formally $g(x)$ should satisfy $f(x) \neq 0 \rightarrow g(x) > 0$. To analyze the convergence of an MCMC estimator, we define $n\,\mathrm{SE}_j$ as the product of the number of all the samples n and the square error of pixel j:

$$n\,\mathrm{SE}_j = n(\hat{I}_j - I_j)^2, \tag{5}$$

where $\hat{I}_j$ is the MCMC estimated value of I_j (Tab. 1). Then, for the whole image, nMSE will be the average of the $n\,\mathrm{SE}_j$ across all the pixels. In the remaining section, we focus on one pixel $n\,\mathrm{SE}_j$ which can be easily applied to the full image to get nMSE.

We will show that for a valid MCMC estimator, the expectation of $n\,\mathrm{SE}_j$ converges to a constant determined by the variance and correlation of the samples. This constant also measures how fast the MCMC estimator converges to the ground truth. Specifically, we express the expected value of $n\,\mathrm{SE}_j$ by adopting Eq. (5) and the definition of $\hat{I}_j$, we have:

$$\mathrm{E}[n\,\mathrm{SE}_j] = \mathrm{E}\left[n\left(\frac{\sum_{s=1}^{n} w_j(x_s)}{n} - I_j\right)^2\right] \tag{6}$$

$$= \mathrm{E}\left[\frac{\left(\sum_{s=1}^{n}(w_j(x_s) - I_j)\right)^2}{n}\right]. \tag{7}$$

By expanding the sum of squares, we can rewrite this expression as:

$$\mathrm{E}[n\,\mathrm{SE}_j] = \frac{1}{n}\sum_{s=1}^{n}\sum_{t=1}^{n}\mathrm{E}[(w_j(x_s) - I_j)(w_j(x_t) - I_j)] \tag{8}$$

which can be decomposed into:

$$\begin{aligned}\mathrm{E}[n\,\mathrm{SE}_j] = &\frac{1}{n}\sum_{s=1}^{n}\mathrm{Cov}[w_j(x_s), w_j(x_s)] \\ &+ \frac{2}{n}\sum_{s=1}^{n}\sum_{k=1}^{n-s}\mathrm{Cov}[w_j(x_s), w_j(x_{s+k})],\end{aligned} \tag{9}$$

where the first term, $\mathrm{Cov}(w_j(x), w_j(x))$, is the variance of the distribution of samples, which is the variance of the target distribution, and we denote it as $\mathrm{Var}(w_j(x))$. The latter term is the correlation between these samples. For simplicity, we introduce $R_k = \mathrm{Cov}(w_j(x_s), w_j(x_{s+k}))$ to represent the correlation between samples from the stationary distribution that are k steps apart. Then,

Algorithm 1 Estimation of L_j

```
Set T and k_max
L̂_j ← 0
for i = 1,2, ..., T do
    Sample x_0 ~ u(x)
    S ← h_j^2(x_0) f^2(x_0) / (g(x_0) u(x_0)) − I_j^2
    for k = 1,2, ..., k_max do
        Sample x_k ~ M(x|x_{k−1})
        S += 2 h_j(x_0) h_j(x_k) f(x_0) f(x_k) / (g(x_k) u(x_0)) − 2 I_j^2
    end for
    L̂_j += S / T
end for
Return L̂_j
```

Eq. (9) can be simplified into:

$$\mathrm{E}[n\,\mathrm{SE}_j] = \underbrace{\mathrm{Var}(w_j(x))}_{\textbf{Variance Term}} + \underbrace{2\sum_{k=1}^{n}\frac{n-k}{n}R_k}_{\textbf{Covariance Term}} \tag{10}$$

$$= \mathrm{Var}(w_j(x)) + 2\sum_{k=1}^{n} R_k - \frac{2}{n}\sum_{k=1}^{n} kR_k. \tag{11}$$

This indicates that the convergence of SE_j is jointly determined by the variance of the target distribution and the covariance between the samples. In the appendix, we demonstrate that R_k decays geometrically as k increases. Consequently, the sum of R_k converges to a constant. The product kR_k forms an *Arithmetico-geometric sequence* whose infinite sum is also convergent. After scaling by $\frac{2}{n}$, this sum vanishes as n tends to infinity. Therefore, $n\,\mathrm{SE}_j$ converges to a constant L_j:

$$L_j := \lim_{n\to+\infty}\mathrm{E}[n\,\mathrm{SE}_j] = \mathrm{Var} + 2\sum_{k=1}^{+\infty} R_k, \tag{12}$$

where $w_j(x)$ is omitted for simplicity. We denote L_j as the convergence constant of pixel j. Additionally, with $\frac{R_k}{\mathrm{Var}}$ representing the k-lag correlation of the samples, we can rewrite our conclusion as:

$$\lim_{n\to+\infty}\mathrm{E}[n\,\mathrm{SE}_j] = \mathrm{Var}\left(1 + 2\sum_{k=1}^{+\infty}\frac{R_k}{\mathrm{Var}}\right). \tag{13}$$

In statistics literature, the term $\tau = 1 + 2\sum_{k=1}^{+\infty}\frac{R_k}{\mathrm{Var}}$ is also called the *autocorrelation* term, and n times its reciprocal is usually referred to as the Efficient Sample Size: $n_{ess} = \frac{n}{\tau}$. Existing statistical studies also show that for a valid MCMC estimator, its Monte Carlo standard error is proportional to $\frac{1}{\sqrt{n_{ess}}}$, due to the correlation between the n samples [DDFG*01].

4.2. Monte Carlo based convergence estimator

The above analysis reveals the convergence property with L_j. However, for a specific scene and a specific MCMC render, there is no analytical solution to get L_j directly. As a result, we further derive a Monte Carlo (MC) estimator for the convergence coefficient L_j.

The variance term $\mathrm{Var}(w_j(x))$ is relatively simple and can be

expressed as:

$$\mathrm{Var}(w_j(x)) = \int_\Omega \frac{h_j^2(x) f^2(x)}{g(x)}\,\mathrm{d}x - I_j^2, \tag{14}$$

which can be estimated by sampling x_0 from a uniform distribution $u(x)$ and computing the MC estimates of its variance.

To estimate the covariance term, we start by estimating the k-lag correlation R_k. Based on the definition, if we have a k-length Markov chain, the correlation between its first sample x_0 and the last sample x can be expressed as:

$$R_k = \int_\Omega \int_\Omega \frac{h_j(x_0) f(x_0) h_j(x_k) f(x_k)}{g(x_0)} M^k(x_k, x_0)\,\mathrm{d}x_0\,\mathrm{d}x_k - I_j^2, \tag{15}$$

where $M^k(x_k, x_0)$ is the k-step transition probability ($x_0 \to x_k$). For each given k, we can sample x_0 from a uniform distribution $u(x)$ in the primary space and compute the k-order correlated sample x_k following the Markov process. Each sampled k-length Markov chain can now be regarded as an *MC-sample*, following the joint distribution of:

$$p(x_0, x_k) = u(x_0) M^k(x_k, x_0),$$

so that the estimate of R_k can be rewritten as:

$$\mathrm{E}[\hat{R}_k] = \int_\Omega \int_\Omega p(x_0, x_k) \hat{R}_k\,\mathrm{d}x_0\,\mathrm{d}x_k = R_k.$$

Thus, we can compute the MC estimate of R_k based on the sampled x_0, x_k, and $u(x_0)$ with:

$$\hat{R}_k = \frac{h_j(x_0) f(x_0) h_j(x_k) f(x_k)}{g(x_0) u(x_0)} - I_j^2. \tag{16}$$

where I_j^2 is the ground truth pixel value. In practice, we run two independent BDPT renders of the image and compute the product to get an estimate of the I_j^2 term.

We then need to compute the sum of an infinite series of R_k to evaluate the covariance term. In the appendix, we provide proof that the sum of R_k converges *exponentially* to a constant as k increases, and its convergence rate is determined by the eigenvalues of M. Thus, a higher order R_k will have a diminishing contribution to the final estimate. In practice, we can take a predefined threshold k_{max} without introducing significant estimation bias or use Russian Roulette to determine k stochastically. The detailed algorithm for the MC estimator for both the variance and the correlation term is illustrated in Alg. 1. Note that the k-length Markov chain only needs to be sampled once for each x_0 to compute all R_k with k from 1 to k_{max}.

In the supplementary material, we will show a simple 1D example where the variance and covariance terms can be accurately predicted by the theory with the analytically solved eigenvectors and eigenvalues, and show that our MC-estimator can well match the theoretical prediction.

Rejection sampling. Notice that the primary computational cost for the MC-estimator in Alg. 1 is evaluating a Markov chain with k samples starting from every sampled x_0. From Eq. (16), we can observe that every sample in this Markov chain has its contribution to L_j proportional to $f(x_0)$. Thus, making $x_0 \sim u(x)$ proportional to $f(x)$ will improve the efficiency of the MC-estimator. As a result, we adopted a rejection sampling scheme to determine whether we evaluate the entire Markov chain starting with x_0.

Algorithm 2 Estimation of L with rejection sampling

```
Set T and k_max
L̂_j ← 0
for i = 1,2, ..., T do
    Sample x_0 ~ u(x), r_{x_0} ~ [0,1]
    p_{x_0} ← min(f_j(x_0), f_max) / f_max
    if r_{x_0} < p_{x_0} then
        S ← h_j^2(x_0) f^2(x_0) / (g(x_0) u(x_0)) − I_j^2
        for k = 1,2, ..., k_max do
            Sample x_k ~ M(x|x_{k−1})
            S += 2 h_j(x_0) h_j(x_k) f(x_0) f(x_k) / (g(x_k) u(x_0)) − 2 I_j^2
        end for
        L̂_j += S / (p_{x_0} T)
    end if
end for
Return L̂_j
```

Specifically, we first use a uniform sampling in the primary space to roughly estimate $\hat{f}_{max}$, the maximum value of $f(x)$. Then for each sampled x_0, we first evaluate $f(x_0)$ and reject the sample with a probability of $\min(1, \frac{f(x_0)}{\hat{f}_{max}})$. Although this rejection sampling scheme will waste computation in evaluating $f(x_0)$, it saves a large amount of computation by avoiding evaluating a long Markov chain which will have little contribution to the estimation of L_j due to the low contribution of the initial path x_0. Since the goal of rejection sampling is to exclude initial samples with low contributions, which do not need a precise f_{max}. An underestimated $\hat{f}_{max}$ should be sufficient. The detailed algorithm for the MC-estimator with rejection sampling is described in Alg. 2.

5. Validation

To validate the accuracy and efficiency of our proposed convergence estimator, we compare the expected convergence calculated with our theory to the actual MSE measured from real MCMC rendering processes.

Accuracy. To test the accuracy of our method, we chose 3 typical scenes often used in evaluating MCMC rendering algorithms. The *Torus* scene features a dominant object with difficult light transport effects, the *Veach Door* scene requires the rendering algorithm to find challenging long paths, and the *Breakfast Room* scene showcases a more generic rendering scenario with balanced light transport types. For the MCMC rendering algorithm, we chose PSSMLT [KSKAC02] and H2MC [LLR*15] as two examples, and used our MC-based convergence estimator following Alg. 2. In practice, unless otherwise noted, we used $k_{max} = 4096$ and $T = 1024$spp for each evaluation.

To measure the actual MSE, we rendered the same image using different seeds and computed the MSE for each render. We then averaged the MSE over those different rendering trials to get the

expected MSE. Fig. 2 illustrates the convergence plot predicted by our theory and the actual measured MSE. By considering both the variance and covariance, the prediction of our method matches well with the actual measured MSE. Predictions using only the variance term differed significantly from the actual convergence.

For all the experiments, the reference images were rendered with BDPT using $2M$spp, which guarantees that the rendering results are fully converged.

Efficiency. We further evaluate the performance of our estimator and compare it to actually running the MCMC renderer. We tested it on the *Breakfast Room* scene and evaluated it with both the PSSMLT and H2MC algorithms. For our MC-based estimator, we ran it with the same configuration as above; then ran the MCMC renderer for an equal computation time and measured the MSE by comparing it to the reference.

Fig. 3 (a) illustrates the predicted nMSE compared to the equal time computed nMSE using the rendered result of MCMC algorithms. A ground truth nMSE is also provided by running MCMC rendering for a sufficiently long time and averaging out over 256 renders using different seeds. Due to the non-uniform convergence properties of MCMC rendering algorithms, we ran the MCMC render multiple times and plotted multiple curves. The measured nMSE from the MCMC render suffers from severe variance among different runs and diverges from the ground truth nMSE. On the other hand, our prediction already matches the reference with reasonable accuracy.

A conventional way to reduce the variance is by averaging the rendered image of multiple independent MCMC runs with different seeds and then measuring the MSE. Fig. 3 (b)-(d) demonstrates results that measure the nMSE with $4, 16$ and 64 independent runs and compares them with our prediction, with computation time listed. Experimental results show that even averaging with 64 independent MCMC runs, the measured MSE still suffers from non-uniform convergence. As a result, for each configuration, we also perform the "multi-run-average" experiment four times and plot four curves (each curve is already an average of multiple runs) to show the differences between different experiments. Our method takes much less computation time while providing an accurate prediction, whereas the conventional method takes much longer. Even with 64 independent runs, the result nMSE from the conventional method still shows some bias from the ground truth.

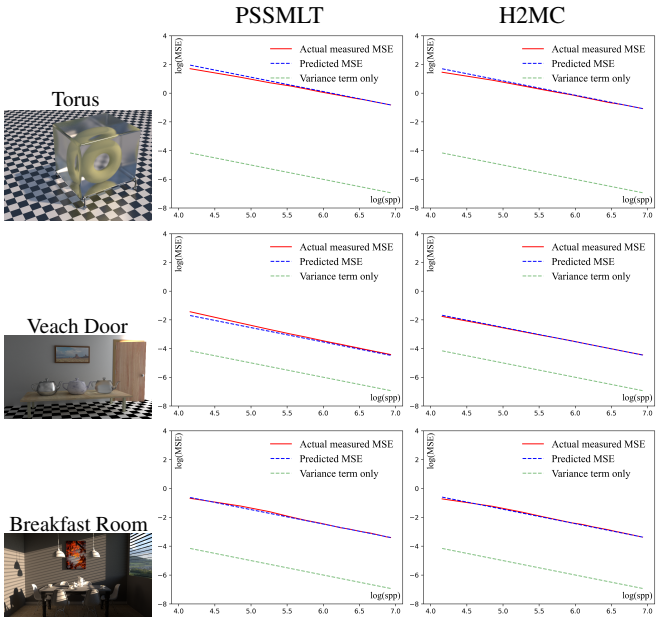

Figure 2: *We validated our MC-based convergence constant estimator on three scenes using two MCMC rendering algorithms. Our predictions matched the actual measured* MSE *in all cases, while predictions using only the variance term significantly differed from the actual convergence.*

Method	spp_e	t_e (min.)	MCMC		Ours	
			SD	RSD	SD	RSD
H2MC	1024	6.4	3.127	9.2%	0.709	2.1%
PSSMLT	1536	5.1	4.572	12.7%	0.689	1.9%

Table 2: *Equal time quantitative evaluation by comparing the standard deviation (SD) and relative standard deviation (RSD) of the estimated* MSE.

Our algorithm in Fig. 3 (a) takes $k_{max} = 4096$, $T = 512$spp, and (f) takes $k_{max} = 2048$, $T = 256$spp. All the performance timings are evaluated on a workstation with a Intel Xeon W-2145 CPU.

As evaluated above, both our method and the "multiple run average" converge to the same estimation of the MSE. We choose the standard deviation of the results of both methods as the evaluation metric for a quantitative assessment, since a lower standard deviation indicates a more stable estimate.

Specifically, we conducted 256 independent MCMC renderings with a fixed spp_e, recording the average rendering time t_e and calculating the standard deviation and relative standard deviation of the results' MSE values. Similarly, we applied our MC-estimator for 256 runs using the same time budget t_e with varying numbers of samples and computed the standard deviation of its estimated MSEs. As shown in Table 2, our method yields a lower deviation compared to estimating with "multiple MCMC runs".

Note that our method also evaluates a MC-estimate of I^2, thus not needing a reference image. However, the MCMC render results need a separately computed reference image to evaluate the MSE. In all the above evaluation, our method includes the computation time of the I^2 term, while all the MCMC renders do not include the BDPT computation time for rendering the reference image.

To analyze the robustness and efficiency of our method, we first plot the estimated nMSE changes as k_{max} increases, taking $T = 2048$spp to rule out randomness. As illustrated in Fig. 4, the estimated nMSE converges exponentially as k_{max} increases, showing that setting a reasonable k_{max} should be sufficient to guarantee a good estimate.

We then set $k_{max} = 4096$ and plot the estimated nMSE with increasing spp for T. Since the computation of our method is proportional to the spp, the spp can also be regarded as a rough time

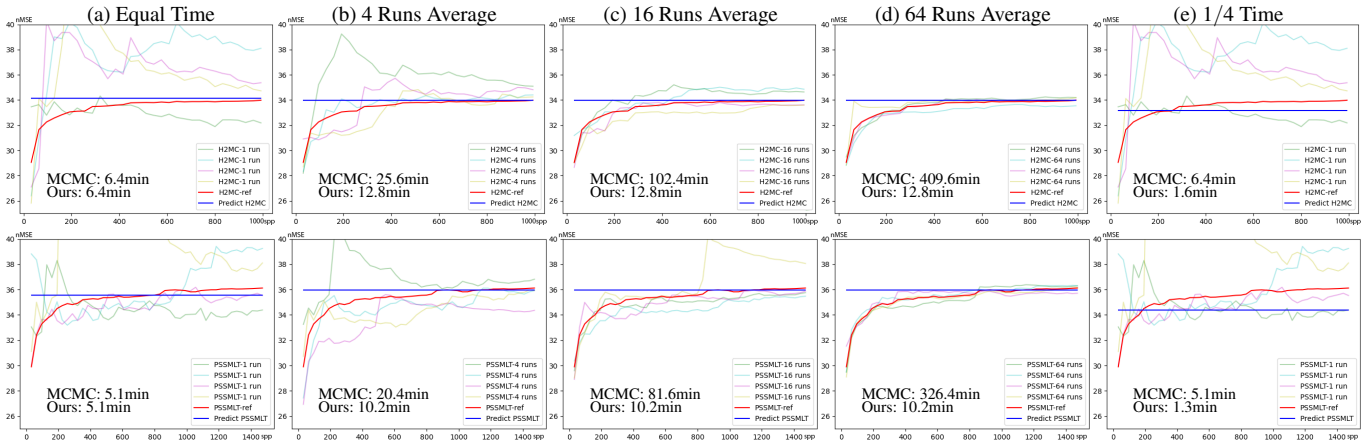

Figure 3: *Our MC-based estimation (blue curve) provides an accurate estimate of the convergence properties (measured with n*MSE*) of a given MCMC rendering algorithm (top-row: H2MC, bottom-row: PSSMLT), which well matches the reference (red curve). With the same computation time (a), the MCMC rendering algorithm suffers from non-uniform convergence, and the actual* MSE *varies significantly in different experiments. Even with much longer computation times, taking averages from multiple independent runs (b)-(d), the MCMC results still have a gap from the reference, which takes* 256 *independent runs to average out the non-uniform convergence effects. (e) With only* $1/4$ *of the computation time, our method already makes a reasonable prediction.*

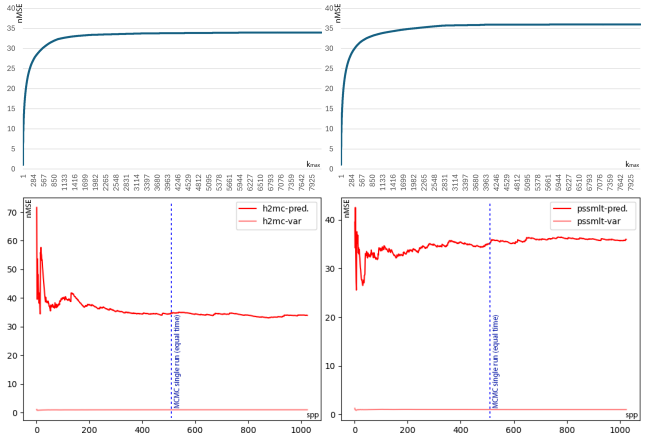

Figure 4: *Robustness and efficiency analysis of our method, on H2MC (left) and PSSMLT (right). The top row plots the estimate of n*MSE *converges exponentially as k_{max} increases. The bottom row plots the estimate of n*MSE *changes as the sampled spp increases. We also draw a dashed line to indicate the computation time which the corresponding MCMC render takes for render a single image. Note that our method already reaches a stable estimate of n*MSE *before MCMC render finishes rendering a single image.*

estimate. As a result, we also marked a dashed line to indicate the time the corresponding MCMC rendering algorithm takes to render a single image (as the curve shown in Fig. 3(a)). Note that our method already reaches a stable estimate of nMSE before the MCMC rendering algorithm finishes rendering a single image, which still suffers from non-uniform convergence. Due to the nature of MC-estimates, it is possible to progressively increase T and K_{max} in order to attain the desired prediction accuracy.

6. Analyzing the Choices of Stationary Distribution

According to Equation (11), the convergence speed of an MCMC rendering algorithm is determined by two factors: the variance of the steady-state distribution and the correlation between the samples. Both are affected by the choice of the stationary distribution of the Markov-chain $g(x)$. In fact, the common choice of setting $g(x) \propto f(x)$ only minimizes the variance term but introduces significant correlation. Existing methods already indicate that altering the target distribution together with a replica exchange scheme [OYH*13, ŠK16, BJ19] can improve the MCMC convergence.

With our efficient convergence estimation, we can *predict* the performance of MCMC with different stationary distributions. In this section, we provide a straightforward roughening scheme that demonstrates how, for specific light transport effects, appropriately modifying $g(x)$ in MCMC might achieve better performance, even without adopting advanced schemes like replica exchange.

Roughening. is a common technique for rendering highly specular surfaces. It makes sampling difficult specular paths easier by increasing the surface roughness. However, naive roughening can introduce bias. Our theory shows that properly designed roughening can achieve unbiased rendering by matching the target distribution. While roughening generally increases variance, it can also reduce sample correlation, potentially balancing variance and correlation for faster convergence.

We create a simple roughening scheme by scaling the BRDF's roughness with a fixed factor S. For dielectric BSDF, we adjust the roughness value, and for Phong models, we scale the power factor. During MCMC rendering, the entire algorithm, including sample mutation and acceptance computation, works within the modified scene. When determining the final contribution to each image pixel, instead of using a fixed value for final pixel contributions, we apply a splat weight function $w_j(x) = \frac{h_j(x)f(x)}{g(x)}$. To calculate this score,

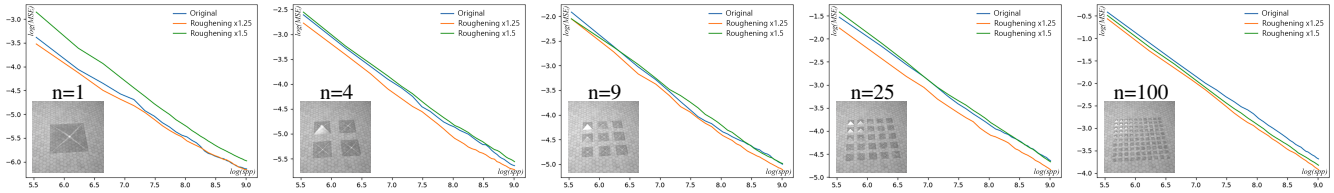

Figure 5: *Our roughening scheme applied to scenes with varying numbers of specular objects.*

we determine $f(x)$ for the original scene and $g(x)$ for the modified scene.

We tested our roughening scheme with three different MCMC rendering algorithms: PSSMLT [KSKAC02], H2MC [LLR*15], and LMC [LZBG20]. Three scenes with complex specular light transports were selected to evaluate the roughening concept.

Figure 6 presents the rendering results of three algorithms using the original configuration and two different roughening scales ($S = 1.25$ and $S = 1.5$). Additionally, a zoomed-in crop is provided to illustrate how challenging purely specular paths were sampled. We also listed the average MSE among those methods measured from 16 independent runs.

We analyze how roughening improves MCMC rendering convergence using our theory, focusing on the *Miniatures Orb* scene with the H2MC rendering algorithm. Using our MC-based convergence estimator, we compute variance and correlation terms, as shown in Figure 1. The figure indicates strong correlation effects due to specular light transport, validated by the low acceptance rate of global mutation proposals. Roughening reduces correlation effects by simplifying specular light transport, though it slightly increases variance. Overall convergence rates remain higher when roughening is properly configured. To obtain a robust estimate for this scene, we use $K_{max} = 8192$, and $T = 2048$spp, due to the high correlation of specular paths. With this configuration, our estimation takes 88 minutes, which is still faster than rendering the image with MCMC (which takes 97 minutes).

We further analyze when roughening is beneficial for MCMC rendering. As illustrated above, roughening specular paths improve global exploration, thus benefiting scenes that require extensive exploration. Figure 5 shows variations of a base scene with different numbers of specular objects. More specular objects create new disjoint path sub-manifolds that need sampling, requiring more global exploration. As shown in Figure 5, experiment results confirm that roughening is more beneficial with an increased number of specular objects. This provides an empirical guideline for choosing proper roughening for the scene. Note that this idea is a preliminary application indicating that selecting a target distribution different from the energy contribution distribution can enhance convergence. Identifying the optimal target distribution is left for future work.

7. Discussion and Conclusion

Performance over MCMC rendering. Our experimental results demonstrate that predicting the MC-estimate of nMSE is faster than having the MCMC rendering algorithm render a single image. The fundamental reason is that our method is specifically designed for estimating the nMSE averaged over the full image efficiently, while the MCMC rendering algorithm is designed to render individual pixels of the full image. Our theory and practice show that, although the sample correlation leads to non-uniform convergence of MCMC, which is hard to predict, its average performance (on multiple independent MCMC runs) can still be estimated efficiently and robustly. We believe our MC-based convergence estimator can be an effective tool for analyzing the performance of future MCMC rendering algorithms.

Previous works on altering the stationary distribution. The majority of previous approaches that modify the stationary distribution have focused on minimizing the relative error, from early attempts [Vea98] to more recent works [ZD20], since it is clear that using energy distribution is not optimal. In terms of minimizing the MSE, using a target distribution proportional to the path contribution is the most common choice. However, as demonstrated in our experiments, even with the MSE metric and using a single Markov chain, this choice is not always optimal. As demonstrated in Sec. 6, our convergence estimation tool can be used to analyze when different target distributions altered by roughening lead to a trade-off between sample variance and correlations. With such analysis, we can design different target distributions for various light paths, yielding better convergence with simple single-chain MCMC, without adopting advanced schemes like [ŠOHK16, BJ19]. This showcases the potential of target function design in MCMC rendering.

Previous works based on roughening. Roughening is a common technique in MCMC rendering for improving convergence speed, but it often introduces bias. To keep the introduced error low while reducing the variance, a set of path space regularization schemes has been proposed [KD13, WDH*21]. With a sufficient number of samples, the regularization can also be progressively relaxed to achieve consistent convergence. Roughening can also be used to create a higher temperature chain in the replica exchange MCMC [ŠK16], which encourages the exploration of the sample space and helps the convergence of the original MCMC chain that still uses the original target distribution. Our roughening scheme aims not to compete with those regularization schemes in terms of convergence speed but to provide a practical example demonstrating that even for the vanilla single-chain MCMC scenario, setting the target distribution to be proportional to the path contribution may not lead to the fastest convergence. We believe that by demonstrating this together with our convergence estimation, future research on designing better target distributions for MCMC rendering can be inspired.

Adaptive MCMC. Our MC-based estimator is specifically designed for low-order Markov chains. Sophisticated algorithms, such as adaptive MCMC that devise a time-varying control function

[AT08], or methods that leverage cache mechanisms [LZBG20], will introduce very long-range correlation effects. Although these effects are still covered by our theoretical analysis, they require more comprehensively designed MC estimators. However, as the sample correlation also remains an important factor for those methods, the general idea of improving the target distribution still holds. As our experimental results show, the roughening application also provides benefits to methods that build on adaptive MCMC like LMC [LZBG20].

Limitations and future work. Our derivations rely on MCMC samples in equilibrium state and only assess the average convergence performance of MCMC algorithms. Specific runs might still diverge from theoretical predictions, and early unstable samples, usually called 'burn-in', are not considered. The number of samples for the MC-based covariance estimator and the choice of k need to be manually specified. An optimal parameter or an adaptive determination of the number of samples, along with the use of advanced samplers for initial samples, presents another direction for future research.

Our rendering-with-roughening application serves as a preliminary example demonstrating that it is possible to balance sample variance and correlation by optimizing the target distribution. A potential area for future research includes pursuing an optimized target distribution for specific rendering tasks. One possible direction involves integrating with a differentiable rendering framework to compute gradients and optimize the target distribution as a hyperparameter.

Conclusion. We present a theoretical framework for estimating the convergence of MCMC rendering algorithms, accounting for both variance and sample correlation. Our Monte Carlo algorithm, developed within this framework, accurately and efficiently estimates MCMC rendering convergence and is validated through examples.

Our analysis shows that setting the Markov chain's stationary distribution proportional to the measurement contribution of light paths may not lead to the fastest convergence. To address this, we incorporate material roughening into MCMC rendering in an unbiased manner. For scenes with complex light-transport effects, a well-designed roughening scheme significantly enhances convergence speed.

Acknowledgments

We thank the anonymous reviewers for their constructive suggestions. We also thank *onmioji* and *Arte Hexe* for providing the 3D models used in the paper.

References

[APSS01] ASHIKHMIN M., PREMOŽE S., SHIRLEY P., SMITS B.: A variance analysis of the Metropolis light transport algorithm. *Computers & Graphics 25*, 2 (2001), 287–294. 2, 3

[AT08] ANDRIEU C., THOMS J.: A tutorial on adaptive MCMC. *Statistics and computing 18* (2008), 343–373. 9

[Ber04] BERG B. A.: *Markov Chain Monte Carlo simulations and their statistical analysis: with web-based Fortran code*. World Scientific Publishing Company, 2004. 3

[BJ19] BITTERLI B., JAROSZ W.: Selectively Metropolised Monte Carlo light transport simulation. *ACM Trans. Graph. 38*, 6 (Nov. 2019). 1, 2, 7, 8

[BJNJ17] BITTERLI B., JAKOB W., NOVÁK J., JAROSZ W.: Reversible jump Metropolis light transport using inverse mappings. *ACM Trans. Graph. 37*, 1 (Oct. 2017). 1, 2

[BRSMD21] BASHFORD-ROGERS T., SANTOS L. P., MARNERIDES D., DEBATTISTA K.: Ensemble Metropolis light transport. *ACM Trans. Graph. 41*, 1 (Dec. 2021). 2

[CA00] CHEN M., ARVO J.: Theory and application of specular path perturbation. *ACM Trans. Graph. 19*, 4 (Oct. 2000), 246–278. 1, 2

[DDFG*01] DOUCET A., DE FREITAS N., GORDON N. J., ET AL.: *Sequential Monte Carlo methods in practice*, vol. 1. Springer, 2001. 3, 4

[FCS22] FANG Y., CAO Y., SKEEL R. D.: Quasi-reliable estimates of effective sample size. *IMA Journal of Numerical Analysis 42*, 1 (2022), 680–697. 3

[GRS95] GILKS W. R., RICHARDSON S., SPIEGELHALTER D.: *Markov Chain Monte Carlo in practice*. CRC press, 1995. 2

[GWH20] GRUSON A., WEST R., HACHISUKA T.: Stratified Markov Chain Monte Carlo light transport. In *Computer Graphics Forum* (2020), vol. 39, Wiley Online Library, pp. 351–362. 3

[Has70] HASTINGS W. K.: Monte Carlo sampling methods using Markov Chains and their applications. 2

[HH10] HOBEROCK J., HART J. C.: Arbitrary importance functions for Metropolis light transport. In *Computer Graphics Forum* (2010), vol. 29, Wiley Online Library, pp. 1993–2003. 2, 3

[HJ11] HACHISUKA T., JENSEN H. W.: Robust adaptive photon tracing using photon path visibility. *ACM Trans. Graph. 30*, 5 (Oct. 2011). 3

[HKD14] HACHISUKA T., KAPLANYAN A. S., DACHSBACHER C.: Multiplexed Metropolis light transport. *ACM Trans. Graph. 33*, 4 (July 2014). 1, 2

[HKD15] HANIKA J., KAPLANYAN A., DACHSBACHER C.: Improved half vector space light transport. *Computer Graphics Forum 34*, 4 (2015), 65–74. 1, 2

[JM12] JAKOB W., MARSCHNER S.: Manifold exploration: a Markov Chain Monte Carlo technique for rendering scenes with difficult specular transport. *ACM Trans. Graph. 31*, 4 (July 2012). 1, 2

[KD13] KAPLANYAN A. S., DACHSBACHER C.: Path space regularization for holistic and robust light transport. *Computer Graphics Forum 32*, 2pt1 (2013), 63–72. 2, 8

[KHD14] KAPLANYAN A. S., HANIKA J., DACHSBACHER C.: The natural-constraint representation of the path space for efficient light transport simulation. *ACM Trans. Graph. 33*, 4 (July 2014). 1, 2

[KKK09] KITAOKA S., KITAMURA Y., KISHINO F.: Replica exchange light transport. *Computer Graphics Forum 28*, 8 (2009), 2330–2342. 2

[KSKAC02] KELEMEN C., SZIRMAY-KALOS L., ANTAL G., CSONKA F.: A simple and robust mutation strategy for the Metropolis light transport algorithm. In *Computer Graphics Forum* (2002), vol. 21, Wiley Online Library, pp. 531–540. 1, 2, 3, 5, 8

[LKL*13] LEHTINEN J., KARRAS T., LAINE S., AITTALA M., DURAND F., AILA T.: Gradient-domain Metropolis light transport. *ACM Trans. Graph. 32*, 4 (July 2013). 2

[LLR*15] LI T.-M., LEHTINEN J., RAMAMOORTHI R., JAKOB W., DURAND F.: Anisotropic Gaussian mutations for Metropolis light transport through Hessian-Hamiltonian dynamics. *ACM Trans. Graph. 34*, 6 (Nov. 2015). 1, 2, 5, 8

[LZBG20] LUAN F., ZHAO S., BALA K., GKIOULEKAS I.: Langevin Monte Carlo rendering with gradient-based adaptation. *ACM Trans. Graph. 39*, 4 (Aug. 2020). 1, 2, 8, 9

[MEL17] MARTINO L., ELVIRA V., LOUZADA F.: Effective sample size for importance sampling based on discrepancy measures. *Signal Processing 131* (2017), 386–401. 3

[MRR*53] METROPOLIS N., ROSENBLUTH A. W., ROSENBLUTH M. N., TELLER A. H., TELLER E.: Equation of state calculations by fast computing machines. *The journal of chemical physics 21*, 6 (1953), 1087–1092. 2

[OHHD18] OTSU H., HANIKA J., HACHISUKA T., DACHSBACHER C.: Geometry-aware Metropolis light transport. *ACM Trans. Graph. 37*, 6 (Dec. 2018). 1, 2

[OKH*17] OTSU H., KAPLANYAN A. S., HANIKA J., DACHSBACHER C., HACHISUKA T.: Fusing state spaces for Markov Chain Monte Carlo rendering. *ACM Trans. Graph. 36*, 4 (July 2017). 1, 2

[OYH*13] OTSU H., YUE Y., HOU Q., IWASAKI K., DOBASHI Y., NISHITA T.: Replica exchange light transport on relaxed distributions. In *ACM SIGGRAPH 2013 Posters*. 2013, pp. 1–1. 2, 7

[Pan17] PANTALEONI J.: Charted Metropolis light transport. *ACM Trans. Graph. 36*, 4 (July 2017). 1, 2

[RCC99] ROBERT C. P., CASELLA G., CASELLA G.: *Monte Carlo statistical methods*, vol. 2. Springer, 1999. 2

[ŠK16] ŠIK M., KŘIVÁNEK J.: Improving global exploration of MCMC light transport simulation. In *ACM SIGGRAPH 2016 Posters*. 2016, pp. 1–2. 2, 3, 7, 8

[ŠK20] ŠIK M., KŘIVÁNEK J.: Survey of Markov Chain Monte Carlo methods in light transport simulation. *IEEE Transactions on Visualization and Computer Graphics 26*, 4 (2020), 1821–1840. 2

[ŠOHK16] ŠIK M., OTSU H., HACHISUKA T., KŘIVÁNEK J.: Robust light transport simulation via Metropolised bidirectional estimators. *ACM Trans. Graph. 35*, 6 (Dec. 2016). 2, 3, 8

[SW86] SWENDSEN R. H., WANG J.-S.: Replica Monte Carlo simulation of spin-glasses. *Physical review letters 57*, 21 (1986), 2607. 2

[Vea98] VEACH E.: *Robust Monte Carlo methods for light transport simulation*. Stanford University, 1998. 2, 8

[VG97] VEACH E., GUIBAS L. J.: Metropolis light transport. In *SIGGRAPH 1997* (USA, 1997), SIGGRAPH '97, ACM Press/Addison-Wesley Publishing Co., p. 65–76. 1, 2

[VGS*21] VEHTARI A., GELMAN A., SIMPSON D., CARPENTER B., BÜRKNER P.-C.: Rank-normalization, folding, and localization: An improved r̂ for assessing convergence of MCMC. *Bayesian analysis 16*, 2 (2021), 667–718. 3

[WDH*21] WEIER P., DROSKE M., HANIKA J., WEIDLICH A., VORBA J.: Optimised path space regularisation. *Computer Graphics Forum 40*, 4 (2021), 139–151. 8

[ZD20] ZIRR T., DACHSBACHER C.: Path differential-informed stratified MCMC and adaptive forward path sampling. *ACM Trans. Graph. 39*, 6 (Nov. 2020). 2, 3, 8

Appendix

Preliminary on Markov Matrix For a given Markov process, we define its state transition matrix as the Markov Matrix $M \in R^{m\times m}$, the Markov process will reaching a steady distribution $|g\rangle$, where $M|g\rangle = |g\rangle$. Here we follows the bra-ket notation for linear algebra, where $|g\rangle$ stands for a column vector and $\langle g|$ as a row vector; $M|g\rangle = |g\rangle$ means matrix M multiply with the vector $|g\rangle$.

Lemma 1 For a valid Markov process with transition matrix $M \in R^{m\times m}$ and steady distribution $|g\rangle$, there must be an eigenvalue of 1, and the magnitudes of all other eigenvalues are smaller than 1.

With lemma 1, we can sort the eigenvalues by their magnitudes decreasing and got $\lambda_s, s = 1,2,...,m$, where $|\lambda_s| \leq 1$,and $\lambda_1 = 1$. The Markov matrix can be written in the diagonal form:

$$M^k = Q\Lambda^k Q^{-1} \tag{17}$$

where $\Lambda^k = Diag_m(1,\lambda_2^k,...,\lambda_m^k)$.

Proof There must be an eigenvalue of 1:

$$M|\vec{g}\rangle = \vec{g}$$

Given any none-zero vector $\vec{u_0}$, we can represent it using the eigenvectors of M:

$$\vec{u_0} = c_1\vec{\beta}_1 + c_2\vec{\beta}_2 + \cdots + c_m\vec{\beta}_m$$

Furthermore, there is

$$A^n|\vec{u_0}\rangle = c_1\lambda_1^n\vec{\beta}_1 + c_2\lambda_2^n\vec{\beta}_2 + \cdots + c_m\lambda_m^n\vec{\beta}_m$$

where $|\lambda_1| \geq |\lambda_2| \geq |\lambda_3| \geq \cdots \geq |\lambda_m|$. On the other hand, since $\vec{g}$ is the stationary distribution, we have

$$\lim_{n\to+\infty} A^n|\vec{u_0}\rangle = [\vec{g},\vec{g},...,\vec{g}]|\vec{u_0}\rangle$$

Thus, $\lim_{n\to+\infty} A^n|\vec{u_0}\rangle$is none-zero and finite. There must be $\lambda_1 = 1, \vec{\beta}_1 = \vec{g}$, and $0 \leq |\lambda_i| < 1$ for $i \geq 2$. □

Lemma 2 The eigenvector corresponding to the eigenvalue 1 is the steady distribution of the Markov process. Thus, we have:

$$Q\Lambda_1 Q^{-1} = \lim_{k\to+\infty} M^k = [|g\rangle, |g\rangle, ..., |g\rangle] =: G \tag{18}$$

where $\Lambda_1 := Diag_m(1,0,0,...,0)$.

Proof

$$\begin{aligned}
&\lim_{k\to+\infty} \Lambda^k = \Lambda_1 \\
&Q\Lambda_1 Q^{-1} = \lim_{k\to+\infty} Q\Lambda^k Q^{-1} \\
&\qquad = \lim_{k\to+\infty} M^k \\
&\qquad = [\vec{g},\vec{g},...,\vec{g}] \\
&\qquad =: G
\end{aligned} \tag{19}$$

□

Proof on the convergence of L_j

Theorem 1 For a valid Markov process, the expectation of $n\,\mathrm{SE}_j$ is converging to a constant L_j:

$$L_j := \lim_{n\to+\infty} n\,\mathrm{E}[(\hat{I}_j - I_j)^2] = \underbrace{\mathrm{Var}(w_j(x))}_{\textbf{Variance Term}} + \underbrace{2\sum_{k=1}^{+\infty} R_k}_{\textbf{Covariance Term}} \tag{20}$$

where:

$$\mathrm{Var}(w_j(x)) = \sum_{x\in\Omega} \frac{h_j^2(x)f^2(x)}{g(x)} - I_j^2 \tag{21}$$

$$R_k = \langle w_j|M^k - G|w_j \circ g\rangle \tag{22}$$

The notion of $\circ$ is the Hadamard product. We further define this constant as the *convergence constant* L_j.

Proof For simplification, we assume that $f(x)$ and $g(x)$ are normalized to $\sum_{x\in\Omega} f(x) = 1$, $\sum_{x\in\Omega} g(x) = 1$. Then for the variance term, based on the definition of mathematical expectation E we have:

$$\mathrm{Var}(w_j(x)) = \sum_{x\in\Omega} ((w_j(x) - I_j)^2 * g(x)) \tag{23}$$

$$= \sum_{x\in\Omega} w_j^2(x)g(x) - I_j^2 \tag{24}$$

substituting with $w_j(x) = f_j(x)/g(x)$ then we get:

$$\mathrm{Var}(w_j(x)) = \sum_{x\in\Omega} \frac{h_j^2(x)f^2(x)}{g(x)} - I_j^2 \tag{25}$$

For the covariance term, we first apply the definition of *Cov* and expand the brackets:

$$R_k = \mathrm{Cov}(w_j(x_s), w_j(x_{s+k})) \tag{26}$$
$$= \mathrm{E}(w_j(x_s)w_j(x_{s+k})) - I_j^2 \tag{27}$$

then according the definition of mathematical expectation E we have:

$$R_k = \sum_{x,y\in\Omega} w_j(x)w_j(y)g(x)M^k(y,x) - I_j^2 \tag{28}$$

We can use the bra-ket notation for matrix-vector multiplication to simplify the above summation, the left term can be written as:

$$\sum_{x,y\in\Omega} w_j(x)w_j(y)g(x)M^k(y,x) = \langle w_j| M^k |w_j \circ g\rangle \tag{29}$$

according to the image formation model, we then expand the I_j^2 as follows:

$$I_j^2 = \sum_{x,y\in\Omega} w_j(x)g(x)w_j(y)g(y) \tag{30}$$

replacing $g(y)$ with its matrix form equivalent $G(y,x)$ and applies the bra-ket notation:

$$I_j^2 = \sum_{x,y\in\Omega} w_j(x)g(x)w_j(y)G(y,x) \tag{31}$$
$$= \langle w_j| G |w_j \circ g\rangle \tag{32}$$

Thus we got the matrix form of the lag-k covariance R_k as follows:

$$R_k = \langle w_j| M^k |w_j \circ g\rangle - \langle w_j| G |w_j \circ g\rangle \tag{33}$$
$$= \langle w_j| M^k - G |w_j \circ g\rangle \tag{34}$$

The variance term ($Var(w_j(x))$) represents the variance of a single sample $w_j(x)$ where x follows the distribution $g(x)$. The infinite series sum of R_k (k=1,2,...) measures the covariance of the sample sequence.

To simplify the summation of R_k in the covariance term, we define some auxiliary diagonal eigenvalue matrix: $\Lambda_1 := Diag_m(1,0,0,...,0)$ and $\Lambda_*^k := Diag_m(0,\lambda_2^k,...,\lambda_m^k)$. With this notation, according to Lemma 2 we can have:

$$\begin{aligned} M^k - G &= Q(\Lambda^k - \Lambda_1)Q^{-1} \\ &= Q\Lambda_*^k Q^{-1} \end{aligned} \tag{35}$$

thus we have:

$$\sum_{k=1}^{+\infty} (M^k - G) = Q\Lambda_s Q^{-1} \tag{36}$$

where Q is the matrix consists of the eigenvectors of M and

$$\Lambda_s = \sum_{k=1}^{+\infty} \Lambda_*^k = diag(0, \frac{\lambda_2}{1-\lambda_2}, \frac{\lambda_3}{1-\lambda_3}, ..., \frac{\lambda_m}{1-\lambda_m}) \tag{37}$$

The *covariance term* can be further simplified as:

$$2\sum_{k=1}^{+\infty} R_k = 2\langle w_j| Q\Lambda_s Q^{-1} |w_j \circ g\rangle \tag{38}$$

From Equ. 38 we can derive that the cumulative lag-k covariance converges exponentially to a constant as k increases. Also according to Equ.25 the variance term is a positive constant value regardless of n, hence the sum of the variance and covariance term converges to a constant:

$$\lim_{n\to+\infty} n\,\mathrm{SE}_j = \mathrm{Var}(w_j(x)) + 2\langle w_j| Q\Lambda_s Q^{-1} |w_j \circ g\rangle$$

□

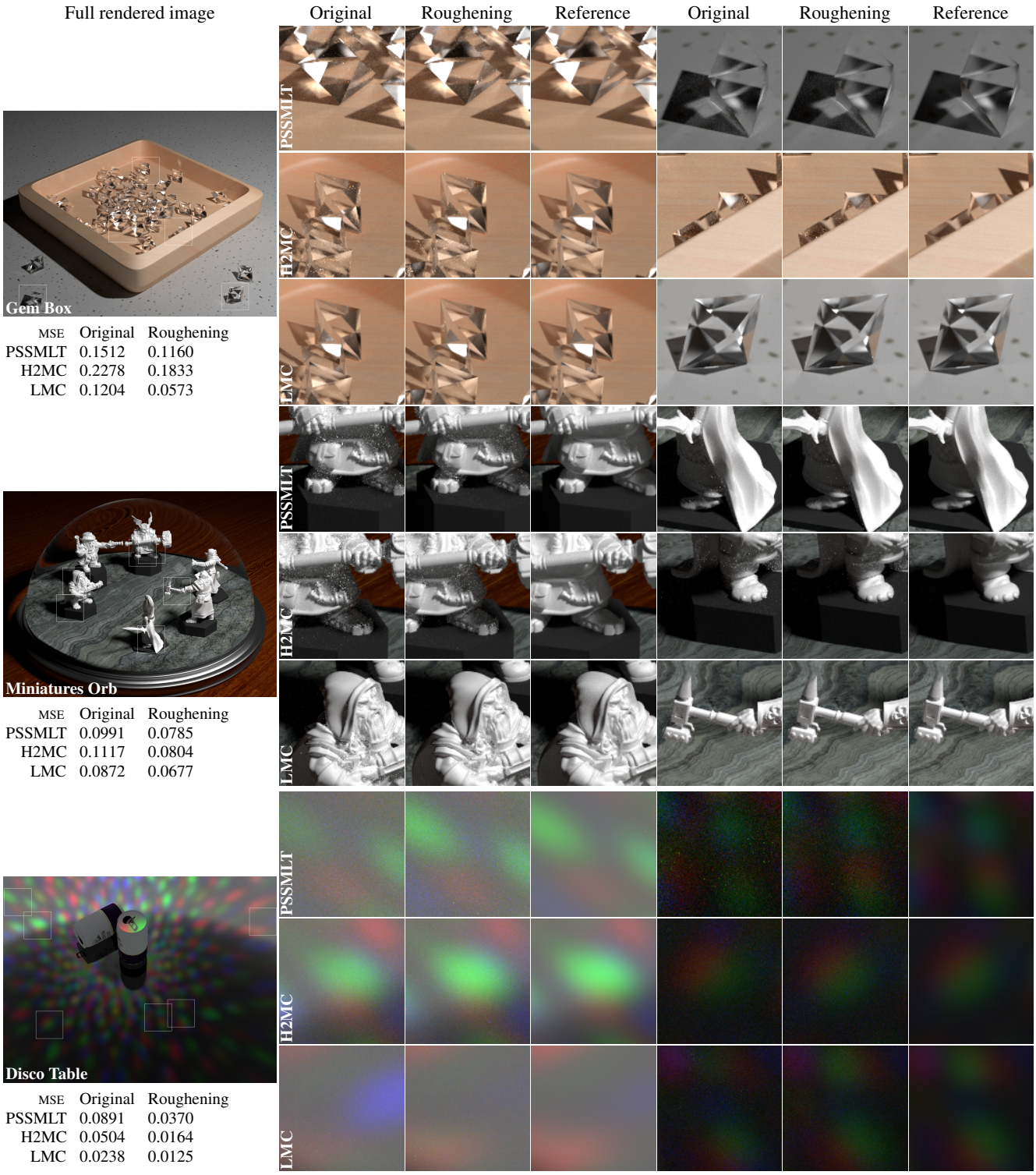

Figure 6: *Equal time comparisons among three MCMC algorithms with and without BRDF roughening. Roughening BRDFs by* 1.25× *significantly improves sampling for specular paths. The inserted table shows the mean squared error (*MSE*), averaged over* 16 *runs to eliminate randomness.*