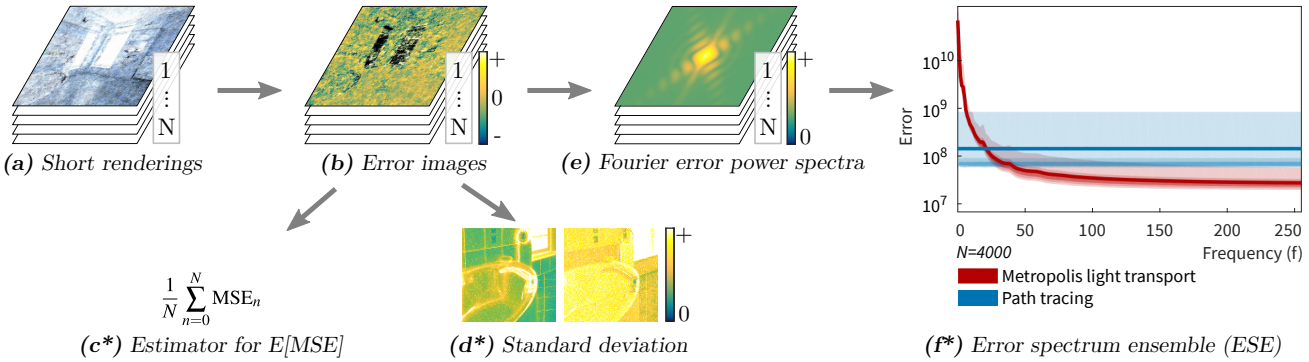
Eurographics Symposium on Rendering 2019
T. Boubekeur and P. Sen
(Guest Editors)

Volume 38 (2019), Number 4

Quantifying the Error of Light Transport Algorithms

A. Celarek[1,2], W. Jakob[3], M. Wimmer[2] and J. Lehtinen[1]

[1]Aalto University, Finland [2]TU Wien, Austria [3]École Polytechnique Fédérale de Lausanne (EPFL), Switzerland



(a) *Short renderings* **(b)** *Error images* **(e)** *Fourier error power spectra*

(c*) *Estimator for E[MSE]* **(d*)** *Standard deviation* **(f*)** *Error spectrum ensemble (ESE)*

Figure 1: *Flow chart of the proposed method with outputs (*): A long rendering process is partitioned into many short runs (a) which are used to estimate error images (b). These are used to calculate a reliable estimate of the expected mean square error (MSE, c*), that could e.g. be used to rank a set of different rendering algorithms. The error images (b) are also used to generate a standard-deviation-per-pixel visualization (d*), which shows which of several competing algorithms is best for a specific lighting situation. Finally, Fourier power spectra (e) are computed and combined into the error spectrum ensemble (ESE, f*) that plots the expected error and outliers with respect to frequency, visualizing for instance correlation between pixels.*

Abstract

This paper proposes a new methodology for measuring the error of unbiased physically based rendering algorithms. The current state of the art includes mean squared error (MSE) based metrics and visual comparisons of equal-time renderings of competing algorithms. Neither is satisfying as MSE does not describe behavior and can exhibit significant variance, and visual comparisons are inherently subjective. Our contribution is two-fold: First, we propose to compute many short renderings instead of a single long run and use the short renderings to estimate MSE expectation and variance as well as per-pixel standard deviation. An algorithm that achieves good results in most runs, but with occasional outliers is essentially unreliable, which we wish to quantify numerically. We use per-pixel standard deviation to identify problematic lighting effects of rendering algorithms. The second contribution is the error spectrum ensemble (ESE), a tool for measuring the distribution of error over frequencies. The ESE serves two purposes: It reveals correlation between pixels and can be used to detect outliers, which offset the amount of error substantially.

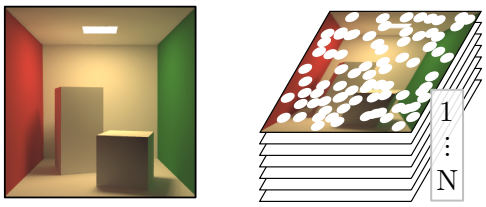
CCS Concepts

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

1. Introduction

Physically based rendering algorithms rely on numerical integration techniques, which yield approximations that converge to an exact solution with increasing computation budget. When limited to a finite computational budget, the inherent rendering error of such algorithms not only differs in magnitude, but also in terms of their spectral characteristics. For instance, Monte Carlo (MC) algorithms produce white noise error spectra, while correlations in Markov chain Monte Carlo (MCMC) methods produce a disproportionate amount of error in low frequency bands. Surprisingly, there is no established standard method for measuring error across such methods, which limits our ability to compare them systematically.

A common approach is to first compute a reference solution, which is then used to calculate the error image for a representa-



(a) *Normal* **(b)** *Proxy*

Figure 2: *We modify the original algorithm (a) by partitioning the rendering process into N shorter equal-time runs that each produce an image (b). The computation budget of those short renderings is typically a few CPU seconds. From the resulting data, we are able to create statistics for any error metric along with per-pixel standard deviation or variance.*

tive run of each algorithm, using an equal rendering budget to make them comparable. Quantitative comparisons are then conducted using mean square error (MSE), or other derived quantities, e.g., root of MSE (RMSE) or other single value metrics. Such error statistics are of limited use with regard to the nature and visual impression of the error. Moreover, they are not informative when assessing the reliability for a particular combination of scene and algorithm. In particular, issues arise from outliers, i.e., low-probability events that introduce a significant amount of error. This can affect individual pixels or larger correlated regions when working with MCMC methods. Frequently, the measurements are supplemented by visual evaluation of the example renderings. However, such qualitative evaluations are inherently subjective and imprecise.

To address these problems, we propose several tools based on a simple *proxy* algorithm (Section 3). Instead of one long rendering, the proxy algorithm takes the average of N short renderings with the same overall budget (Figure 2). If the original algorithm is unbiased, then the resulting proxy is unbiased as well. If the original algorithm has finite per-pixel variance in addition, then it also fulfills the requirements for the central limit theorem (CLT). Hence, its asymptotic convergence rate is $\Theta(1/\sqrt{N})$ in standard deviation and it is consistent. That by itself can be an improvement, as a convergence rate of $\Theta(1/\sqrt{N})$ is not necessarily achieved by every algorithm.

More importantly, the proxy algorithm allows to

- estimate the *expected* MSE and its variance, or other measures of reliability,
- compute standard deviation or variance per pixel images, showing which light effects are problematic, and
- provide more advanced tools based on the short renders.

As one such advanced tool we propose the error spectrum ensemble (ESE). Based on the Fourier transform of the error images, it reveals the spatial frequency content of error as well as outliers. An application of the ESE can be quick the deduction of the magnitude and spatial extent of the correlation between pixels, or of the spatial size and intensity of outliers.

2. Related Work

A number of researchers have investigated the properties of error in light transport algorithms. Arvo et al. [ATS94] identified three sources of error (input data, discretization, and computation) and provided bounds. In contrast to their work, we are only concerned with precise estimation of numerical error in competing integration procedures. Szirmay-Kalos et al. [SKDP99] analyzed the start-up bias and convergence of MLT, but a lot of their analysis is based on very simple example scenes and does not provide a framework for comparing MC with MCMC methods. Ashikhmin et al. [APSS01] showed that the per-pixel variance of Metropolis light transport (MLT) [VG97] is bounded by $\Theta(1/N)$, which is an important insight, but it does not help in quantifying the error. Rousselle et al. [RKZ12] use two buffers and the sample variance to estimate the error in path tracing. This is comparable to our proxy using $N = 2$, but their work is focused on adaptive sampling and not a general error metric. Subr and Kautz [SK13] analyzed the error caused by various Monte Carlo (MC) sampling patterns via Fourier analysis, but their conclusions do not generalize. Whittle et al. [WJM17] provided an extensive overview of error metrics for images, and they examined the influence of poor references for computing those measures. However, they did not investigate their variance, or sensitivity to outliers.

Durand et al. [DHS*05] investigated the frequency content of radiance and how it is influenced by various light-transport phenomena. Zirr et al. [ZAD15] studied possibilities for the visualization and editing of structured light-transport effects. Both of these works study the effect of various physical phenomena, while our work targets statistical estimators.

Kettunen et al. [KMA*15] used the Fourier spectrum to analyze the superior convergence behavior of gradient domain path tracing. Similarly to the proposed ESE, Lehtinen et al. [LKL*13] used radial averages and the Fourier transform to plot error against frequency. However, they used only one long rendering, and neither studied outliers nor the overall reliability. Several other papers also used radial averages of data derived from the Fourier transform for the analysis of low-dimensional sampling patterns [LD08; SK13; PSC*15].

3. Proxy Algorithm

Basics. We treat rendering as a stochastic process that estimates the path-space integral:

$$\langle I_m \rangle_N \approx I_m = \int_\Omega f_m(\bar{x}) \mathrm{d}\bar{x}, \tag{1}$$

where $\langle I_m \rangle$ denotes the estimator or rendering algorithm computing the value of a pixel m, N is the computation budget, Ω refers to the set of all transport paths and f_m to the contribution function.

Throughout the text, angular brackets $\langle X \rangle_Y$ refer to an estimator X (for instance the MSE) related to a particular a rendering algorithm that uses a computation budget of Y. In practice, Y will be proportional to the number of sample evaluations performed by the MC or MCMC scheme.

We limit ourselves to algorithms with finite per-pixel variance that are unbiased (with one exception highlighted in Section 4.3).

The condition on variance is unproblematic as algorithms with infinite variance would not be useful in practice. Unbiasedness is defined as

$$\mathrm{E}\left[\langle I\rangle_N - I\right] = 0, \tag{2}$$

i.e., for any algorithm $\langle I\rangle_N$, the expected value of the algorithm matches the path-space integral..

Consistent algorithms satisfy the following criterion:

$$\lim_{N\to\infty} \langle I\rangle_N = I. \tag{3}$$

Note that not every unbiased algorithm is consistent (and vice versa).

Proxy Algorithm. We define the result of the proxy algorithm as the average of N short and equal-budget renderings of the original algorithm:

$$\langle I'\rangle_N = \frac{1}{N}\sum_{n=1}^{N} \langle I^n\rangle_1 \tag{4}$$

$\langle I^n\rangle_1$ is one of the N independent short renderings (Figure 2).

Central Limit Theorem. Looking at individual pixels, the preconditions of the central limit theorem (CLT) are satisfied independently of the rendering algorithm:

- The short renderings are independent.
- Corresponding pixels share common probability distributions, because they were generated by the same scene, camera and algorithm setup, only using different random numbers.
- Their expectations are defined.
- Their variance is finite.

Due to the CLT, the per-pixel value distributions in the proxy result ($\langle I\rangle_N$, Equation 4) tend to a normal distribution as N grows. The distributions have I_m mean and a standard deviation that varies from pixel to pixel. We therefore propose to routinely show standard deviation per-pixel images (SDpp) (Figure 1e) to visualize the convergence speed throughout an image. Any existing correlation between the pixels, a characteristic property of Markov chain based methods, is not reduced as N grows. Therefore, it would be useful to visualize and measure that correlation – however, this task is non-trivial and left for future work.

Convergence Rate. A direct consequence of the CLT is the asymptotic convergence rate of $\Theta(1/\sqrt{N})$ as measured by per-pixel standard deviation. This is an improvement over the original algorithm if $\Theta(1/\sqrt{N})$ cannot be easily achieved using the built-in parameters. An example is energy redistribution path tracing (ERPT) [CTE05], where quality and rendering budget are influenced by chain length, number of seed paths, and number of chains. The number of runs N, on the other hand, is a parameter of the proxy algorithm. In situations where methods for a better convergence rate are known, e.g., sampling patterns for low dimensions [PSC*15], the proxy algorithm will become a limiting factor. Section 4.3 shows how methods based on the proxy algorithm can be adapted in such cases, and Section 5.2 shows examples of the performance impact in case of Markov-chain-based methods.

Unbiased and Consistent. The proxy algorithm is unbiased, because

$$\mathrm{E}\left[\langle I'_m\rangle_N - I_m\right] = \frac{1}{N}\sum_{n=1}^{N} \mathrm{E}\left[\langle I^n_m\rangle_1 - I_m\right], \tag{5}$$

where n is the index of the short rendering, for every pixel m, which is a consequence of the unbiasedness of the original algorithm (Equation 2). The applicability of the CLT also means that the weak law of large numbers applies, therefore our proxy is not only unbiased, but also consistent. This is an improvement for algorithms that are unbiased, but not consistent.

In the rest of this article we will assume the original algorithm for the short renderings and the proxy for the result and drop the prime symbol ($\langle I\rangle_N = \langle I'\rangle_N$).

Rendering Error. Commonly used error measures for a rendering $\langle I\rangle_K$ include the signed absolute error $\langle \mathcal{E}\rangle_K = \langle I\rangle_K - I$ or the relative error $\langle \mathcal{E}'\rangle_K = (\langle I\rangle_K - I)/(I+\varepsilon)$, where ε is a small offset that prevents division by zero. The absolute error can become large in brightly lit areas, despite such errors likely being imperceptible to a human observer. In contrast, the relative error has a singularity in dark areas that is only ameliorated by the epsilon term. Since scenes with almost black areas are common, we deem the problem of singularity more severe and use the absolute error throughout the paper.

Since I is unknown, it must be estimated. One option is to estimate it independently using a very high rendering budget, i.e., create a ground truth reference image. In our case it is more efficient to use the proxy result $\langle I\rangle_N$ for estimating the error of the individual short renderings $\langle I\rangle_1$:

$$\langle \mathcal{E}_n\rangle_1 = \langle I^n\rangle_1 - \langle I\rangle_N. \tag{6}$$

Such an estimation of signed error is unbiased as $\mathrm{E}[\langle I^n\rangle_1] = \mathrm{E}[\langle I\rangle_N]$. The precision of individual observations of $\langle \mathcal{E}_n\rangle_1$ grows with rising N. Estimating the per-pixel variance requires Bessel's correction to produce an unbiased result.

3.1. Improvements for Measuring Mean Square Error

In classical statistics, MSE is a scalar estimator property. The statistical definition using the estimator from Equation 1 is

$$\mathrm{MSE}(\langle I_m\rangle_K) = \mathrm{E}\left[(\langle I_m\rangle_K - I_m)^2)\right], \tag{7}$$

for only one pixel, and with unknown I_m. In case of unbiased estimators, the statistical MSE is equivalent to (per-pixel) variance.

In contrast, MSE in the context of rendering is often defined as a random variable averaging squared errors over the whole image, calculated after a certain rendering budget K:

$$\mathrm{MSE}(\langle I\rangle_K) = \frac{1}{M}\sum_{m=0}^{M} \left(\langle I_m\rangle_K - I_m\right)^2. \tag{8}$$

So while "mean" stands for "expectation" in statistics, it is used for "average over pixels" in rendering.

Traditional and Proposed MSE. The expectation $\mathrm{E}[\mathrm{MSE}(\langle I\rangle_K)]$ is useful to rank a set of algorithms. This quantity is traditionally estimated (denoted by $\widehat{\mathrm{MSE}}$) using a single run of a rendering algorithm:

$$\begin{aligned}\mathrm{E}\left[\mathrm{MSE}(\langle I\rangle_N)\right] &\approx \widehat{\mathrm{MSE}}_1(\langle I\rangle_N)\\ &= \frac{1}{M}\sum_{m=0}^{M}\left(\frac{1}{N}\sum_{n=0}^{N}\langle I_m\rangle_1 - I_m\right)^2. \end{aligned} \tag{9}$$

Problems can arise due to high variance, which causes the estimates to be far from the true expectation. We propose to use N observations instead:

$$\begin{aligned}\mathrm{E}\left[\mathrm{MSE}(\langle I\rangle_1)\right] &\approx \widehat{\mathrm{MSE}}_N(\langle I\rangle_1)\\ &= \frac{1}{(N-1)M}\sum_{n=0}^{N}\sum_{m=0}^{M}\left(\langle I_m^n\rangle_1 - \langle I_m\rangle_N\right)^2, \end{aligned} \tag{10}$$

where N is the rendering budget of the proxy algorithm, $N-1$ accounts for the Bessel correction, M is the pixel count, $\langle I_m^n\rangle_1$ the value of pixel m in proxy rendering n and $\langle I_m\rangle_N$ pixel m of the proxy algorithm result. In Equation 9, positive and negative errors cancel each other out in the inner sum of the traditional method, which is avoided in the proposed calculation.

Analogy to Statistical MSE. By the law of large numbers, we know that

$$\begin{aligned}\mathrm{E}\left[\mathrm{MSE}(\langle I\rangle_1)\right] =& \widehat{\mathrm{MSE}}_\infty(\langle I\rangle_1)\\ =& \frac{1}{M}\sum_{m=0}^{M}\lim_{N\to\infty}\left(\frac{1}{(N-1)}\sum_{n=0}^{N}\left(\langle I_m^n\rangle_1 - \langle I_m\rangle_N\right)^2\right)\\ =& \frac{1}{M}\sum_{m=0}^{M}\mathrm{Var}\left(\langle I_m\rangle_1\right). \end{aligned} \tag{11}$$

In case of unbiased algorithms, the proposed method converges to the average per-pixel variance as the term within the limes brackets is simply a variance estimate: If considering biased algorithms (Section 4.3), $\langle I_m\rangle_N$ should be replaced by a high quality reference solution and Bessel's correction removed. In both cases, the proposed MSE converges towards the average of the classic statistical MSEs defined in Equation 7.

Rendering Budget Scaling. Due to the CLT,

$$\mathrm{Var}\left(\langle I_m\rangle_K\right) = \frac{\mathrm{Var}\left(\langle I_m\rangle_1\right)}{K}$$

applies for every individual pixel. Since the expectation of the proposed MSE is a linear combination of variances (Equation 11), we deduce for the whole image that

$$\mathrm{E}\left[\mathrm{MSE}(\langle I\rangle_K)\right] = \frac{\mathrm{E}\left[\mathrm{MSE}(\langle I\rangle_1)\right]}{K}.$$

After plugging in the proposed estimator $\widehat{\mathrm{MSE}}_N(\langle I\rangle_1)$ for $\mathrm{E}[\mathrm{MSE}(\langle I\rangle_1)]$ we have a derived estimator for $\mathrm{E}\left[\mathrm{MSE}(\langle I\rangle_K)\right]$

$$\mathrm{E}\left[\mathrm{MSE}(\langle I\rangle_K)\right] \approx \frac{\widehat{\mathrm{MSE}}_N(\langle I\rangle_1)}{X}, \tag{12}$$

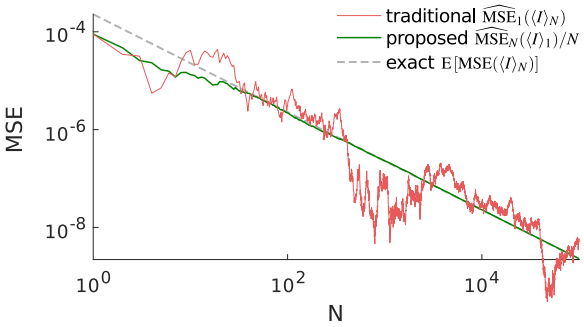


Figure 3: *Comparison of the traditional and the proposed MSE computation for a simple MC integration of Equation 14. The traditional method does not converge to its expectation even with a large computation budget, which the proposed method addresses.*

which will be used for a toy example in the following paragraph. Similarly,

$$\begin{aligned}\mathrm{Var}\left(\langle I_m\rangle_1\right) &= K\times\mathrm{Var}\left(\langle I_m\rangle_K\right)\\ &= \mathrm{Var}\left(\langle I_m\rangle_K\times\sqrt{K}\right).\end{aligned}$$

For squared error expectation, it does not matter whether we compute the error for a budget of exactly 1, or for K and then scale by $\sqrt{K}$. We use this fact to accommodate for algorithms with different rendering costs:

$$\langle\mathcal{E}\rangle_1 = \langle\mathcal{E}\rangle_t\times\sqrt{t}, \tag{13}$$

where t is the actual render time or budget of a short rendering and 1 is the *unit budget* used for comparisons.

Simple Experiment. To illustrate these concepts, we perform a simple experiment, where $M=100$ related integrals are computed and elements I_m are analogous to pixels in renderings. The equation for these elements is

$$I_m = \int_{m/M}^{(m+1)/M}\frac{1}{0.01+x}dx, \tag{14}$$

where $m\in[0,M[$.

The goal is to compute a MSE value for a simple MC integration procedure using uniform sampling. Reference values for the I_ms and the exact $\mathrm{E}[\mathrm{MSE}(\langle I_m\rangle_N)]$ were computed using Mathematica. We compare plots of the behavior of the traditional (Equation 9) and proposed computation method (Equations 12 and 10). Using $\widehat{\mathrm{MSE}}_N(\langle I\rangle_1)/N$, we estimate MSE as if a rendering budget of N were used, but with a lower computation cost.

Figure 3 shows the result of the experiment. The same random samples were reused for the traditional and the proposed method. The proposed method converges to the exact solution. Traditional MSE, on the other hand, converges to a normal distribution, where both, expectation and variance are scaled by N. This is only reliable if the MSE variance is low, i.e., when M is large and there are no outliers.

3.2. The Variance of MSE

The variance of the MSE is given by

$$\mathrm{Var}\left(\mathrm{MSE}(\langle I\rangle_K)\right) = \mathrm{Var}\left(\frac{1}{M}\sum_{m=1}^{M}\langle\mathcal{E}_m\rangle_K{}^2\right) = \\ = \frac{1}{M^2}\left(\sum_{m=1}^{M}\mathrm{Var}\left(\langle\mathcal{E}_m\rangle_K{}^2\right) + \sum_{m\neq m'}^{M}\mathrm{Cov}\left(\langle\mathcal{E}_m\rangle_K{}^2, \langle\mathcal{E}_{m'}\rangle_K{}^2\right)\right), \tag{15}$$

where $\mathrm{MSE}\langle X\rangle$ and $\mathcal{E}\langle X\rangle$ are random variables and not observations. The formula for MSE (Equation 8) and summation of variances were used.

In case of independent pixels (MC), the covariance matrix is zero by definition, and hence the MSE variance is $1/M^{th}$ of the average pixel variance. In algorithms with dependent pixels, a positive sum of the covariance matrix indicates that the positive correlation among pixels is stronger than the negative one. This can potentially increase the variance and impair the reliability of MSE estimates for correlated algorithms like MLT.

In practice, we estimate the variance from the N MSE values of the short renderings. Unfortunately, standard methods to compute confidence intervals cannot be used, because the distribution is not normal.

4. Error Spectrum Ensemble

The error spectrum ensemble (ESE) descriptor provides a new way to characterize the frequency-space distribution of the error of unbiased rendering algorithms.

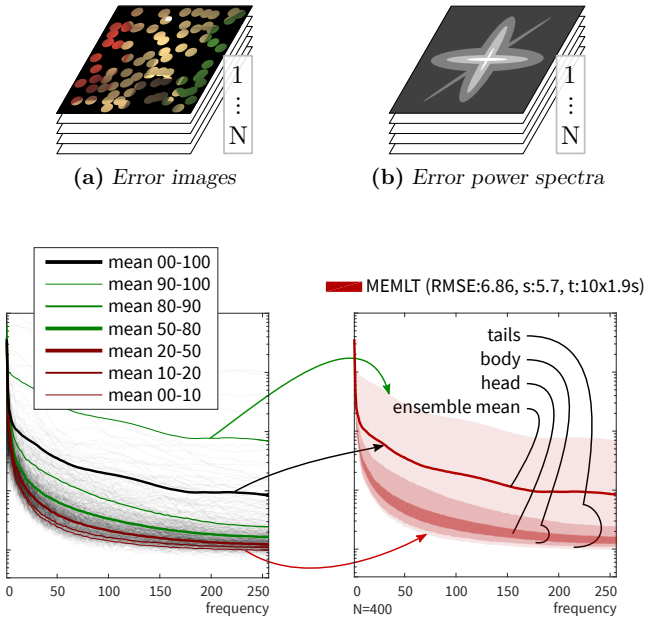
For each of the N renderings, we compute a scaled error image using Equations 6 and 13:

$$\langle\mathcal{E}_n\rangle_1 = \left(\langle I^n\rangle_t - \langle I\rangle_{tN}\right) \times \sqrt{t}, \tag{16}$$

where n is the index of the individual short rendering and t is the rendering time for a single short rendering. We then transform the error images (Figure 4a) into discrete Fourier frequency space and compute associated power spectra by squaring the amplitude of each frequency (Figure 4b). Parseval's theorem applies, i.e., the sum of all power spectrum elements is proportional to the MSE, and we thus interpret the power spectrum as a frequency-dependent MSE statistic.

We reduce the dimensionality of the spectra by computing *radial averages* on concentric rings around the DC term. This allows us to represent every image by a vector of average frequency errors with the DC error at the beginning (Figure 4c). Low radial frequencies represent just a few 2d frequency elements, while higher radial frequencies represent more elements. This means that higher frequencies have a larger contribution to overall MSE and that low frequency radial averages have more variance even if the power spectrum has uniform distribution. Computing the per-frequency mean of those vectors results in a vector that we call the *ensemble mean*. It is a measure of how much error is to be expected on average, per frequency.

Finally, we sort the radial averages of all images according to MSE and divide them into six buckets for each frequency (lowest



(a) *Error images* (b) *Error power spectra*

(c) *Radial averages and percentile means* (d) *Error Spectrum Ensemble*

Figure 4: *For each of the N short renders, we compute an error image, and the associated radial power spectrum. We group the entries of the spectrum into several buckets according to the associated MSE, and use them to form the region borders of tail, body and head of ESE. Additional data is shown in the legend (s = standard deviation of RMSE, t = samples per pixel used × cost for one sample). The ensemble mean does not lie in the head region because of outliers.*

10%, 10-20%, 20-50%, 50-80%, 80-90%, and highest 10%). The per-frequency means of these buckets (Figure 4c) form the borders of the *tail*, *body*, and *head* regions in the descriptor (Figure 4d). Head and body show the typical behavior of the algorithm, while the tail shows outliers.

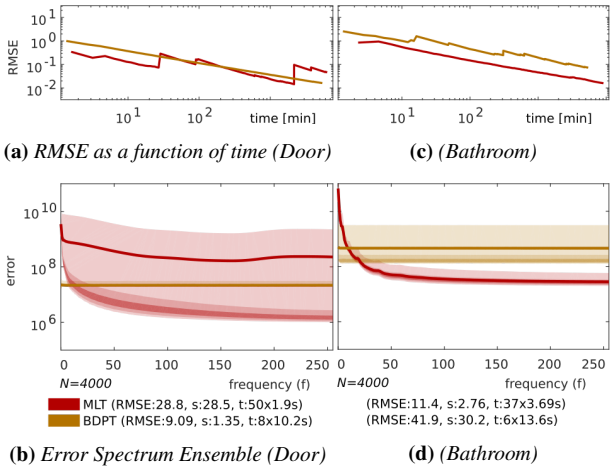
4.1. Implementation

The pipeline starting with rendering commands and ending with the ESE and standard deviation per-pixel images (SDpp) was automated using MATLAB and Python. The rendering time for the short renderings was around 10 single-core seconds on a 2018 AMD Ryzen 7 2700X, and the value of N was 4000 in most tests. An HDR format with 32-bit floating-point was used as 16bit formats cause precision issues when averaging short renderings.

Additional data available from the proxy algorithm is shown in the legend:

- estimated expectation of the RMSE,
- estimated standard deviation of the RMSE,
- computation budget of a short rendering (samples per pixel × average time for a 1-sample-per-pixel image).

We decided to use the linear statistics (RMSE and standard deviation) instead of quadratic ones (MSE and variance) for ease of



(a) *RMSE as a function of time (Door)* **(c)** *(Bathroom)*

(b) *Error Spectrum Ensemble (Door)* **(d)** *(Bathroom)*

Figure 5: *First and second row: Different visualizations of the convergence behavior acquired using the proxy algorithm. Long ESE tails arise due to RMSE jumps because the tails are generated from (R)MSE outliers. Typically, MC algorithms (BDPT) have a flat error spectrum while MCMC ones (MLT) contain more error in low frequencies. Outliers influence the shape of the ensemble mean (b, MLT) and a flat spectral dependence indicates spatially small outliers (d, BDPT).*

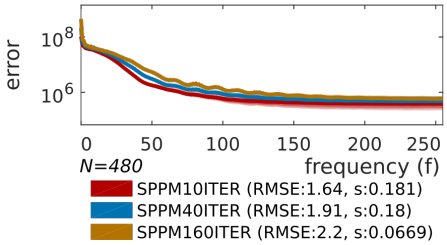


Figure 6: *ESE of the door scene (Figure 7c) for SPPM, which is consistent but biased (T is 10 iterations). The ensemble means perfectly overlap if the convergence rate is $1/T$. In this instance, we see that this is the case for low frequencies. Middle and higher frequencies have a convergence rate below $1/T$, because the $16T$ line is on top.*

interpretation. A full MATLAB implementation is provided in the supplemental material.

4.2. Spectrum Shape and Outliers for Different Algorithms

We briefly discuss the ESE for some common rendering algorithms.

Shape. MC spectra are generally flat, while MCMC spectra contain more error in low frequencies due to correlation (Figures 1g, 5b and 5d). Since the overall brightness in MLT (MCMC) is computed in a separate step, the DC term is independent of all other frequencies. Experiments with the Primary Sample Space MLT algorithm [KSKAC02] (Section 5.3) show that the width of the low-frequency peak decreases as the correlation radius (mutation size) increases, similarly to the Fourier transform of a Gaussian curve. We have also observed that the peak's relative height depends on the amount of correlation between pixels – specifically, it becomes smaller when increasing the percentage of large mutations (which propose an independent path).

In our experiments, we found that the shape of the upper tail border reveals the nature of the underlying outliers: A spectrally flat shape is caused by isolated pixel outliers (Figure 5d, PT), while a peak in low frequencies is a sign of spatially correlated errors (Figure 4d).

Outliers. There is a direct link between the size of the tail area in the ESE and jumps in RMSE over rendering budget plots. RMSE plots (examples in Figures 5a and c) are used to show the asymptotic behavior of algorithms, where a slope of -1 in the logarithmic scaling indicates a convergence rate of $\Theta(1/\text{time})$. The data stems from proxy algorithms with $N = 4000$, where the first X short renderings are averaged to generate the RMSE value at the time point of the duration of X renderings. Using this setup, jumps in the otherwise straight line can be tied to particular short renderings with large error. Those correspond to the largest outliers that go into the computation of the upper tail border, while the bottom tail border corresponds to the values with the smallest error. Hence, the tails' areas correlate with the number and amplitude of jumps (Figures 5b and d). In Figure 5b, we also see that the outliers are the reason for the bad performance of MLT, as most of the samples (the head and body regions) are well below the competing algorithm. Both MC and MCMC algorithms can show this behavior (Figures 5b and d).

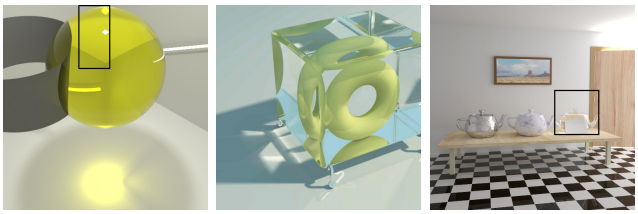
A large tail area, i.e., outliers, not only negatively impacts the performance, but also

- increases the N that is needed for a faithful ensemble mean (in some cases more than the $N = 4000$ that was used),
- slows the convergence of the sample mean of affected pixels to the normal distribution,
- increases the time until SDpp converges sufficiently to provide a good error measure.

To summarize, the ESE is capable of differentiating between algorithms with varying degrees of correlation and varying amount of outliers. That is an improvement over simple measures like MSE and plain visual interpretation of rendering results. However, 'catching' outliers is still a game of luck and therefore, large values for N should be used.

4.3. Adaptive Sampling, Consistent and Sub-$\Theta(1/\sqrt{N})$ Convergence Algorithms

Adaptive sampling can be used to allocate more samples to pixels with a large sample variance, for instance in path tracing. It is not a problem to apply adaptive sampling within the short renderings, as long as the result is unbiased. However, adaptive sampling might influence the overall convergence rate, and the proxy algorithm might decrease the performance of an adaptive sampling algorithm. The situation is similar for sampling patterns (QMC and stratified), which have a better asymptotic convergence rate for the



(a) *Box* **(b)** *Torus* **(c)** *Door*

Figure 7: *Illustration of the figures used in the remainder of our evaluation (tone-mapped). The highlighted regions (a and c) are details shown in Figures 9, 12 and 13, respectively.*

low-dimensional part of the integrand [PSC*15]. The proxy algorithm is not compatible with biased algorithms, even if they are consistent. In these cases, we can use an alternative procedure to inspect the error spectrum and how it is affected by increasing the rendering budget:

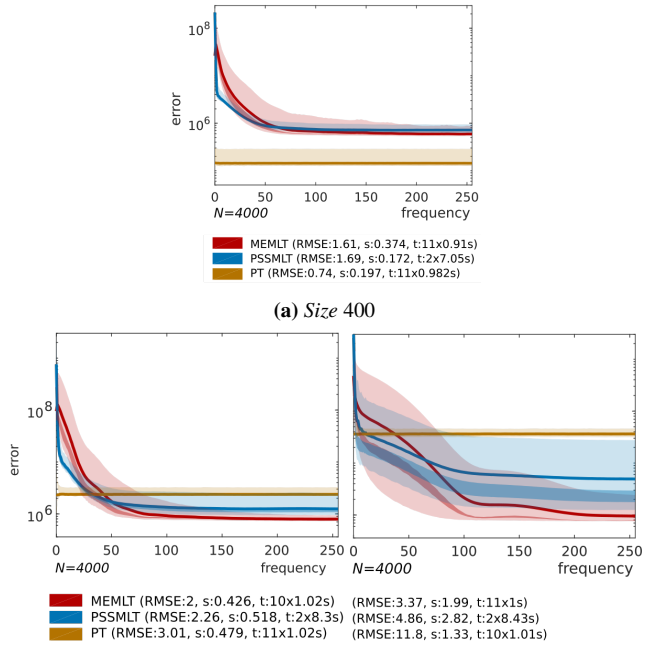
1. Run the algorithm N times with a rendering budget of T, $4T$ and $16T$.
2. Calculate the error images using a reference solution and multiply them with $\sqrt{T}$, $\sqrt{4T}$ and $\sqrt{16T}$, which 'neutralizes' the typical convergence of $\Theta(1/T)$ in squared error.
3. Continue with the steps from Section 4, without rendering cost scaling.

Our reference solutions were computed using a cluster and millions of samples per pixel – several orders of magnitude more than were used for $\langle I \rangle_N$. To put this into perspective, Whittle et al. recommended to use reference images with at least an order of magnitude more samples [WJM17]. If the T, $4T$, and $16T$ curves overlap, then the algorithm behaves like $\Theta(1/\sqrt{T})$ in that frequency and rendering budget range. If curves with larger T are higher, the convergence rate is lower than $\Theta(1/\sqrt{T})$. The same applies to RMSE. A small tail area means that all the runs are approximately the same, i.e., no outliers were found.

Figure 6 shows an example of stochastic progressive photon mapping (SPPM), which is a biased but consistent algorithm. It shows that SPPM converges with a speed of $1/\sqrt{T}$ in low frequencies between 10 and 160 iterations, but with a lower speed for higher frequencies. One would need to run tests with higher iteration counts to see whether the convergence stabilizes, but this becomes more and more expensive in terms of computation resources. The ESE also shows that there are little differences between runs, so it would be a possibility to reduce N and thereby reduce the needed resources. More examples for this procedure, including path tracing with QMC for two scenes, can be found in the supplemental material.

5. Evaluation

We begin with ESE visualizations for various configurations of the box scene (Figure 7a) to provide an intuition about the descriptor. ESE, SDpp, and short rendering examples for more configurations of the box scene, and also several more complex scenes are shown in the supplemental material. The rest of the evaluation will cover



(a) *Size* 400

(b) *Size* 100 **(c)** *Size* 25

Figure 8: *Error descriptors for the box scene with varying light size. The scene becomes increasingly hard to render with smaller light sizes. The impact on overall error is less dramatic for the MLT methods. However, the amount of outliers increases.*

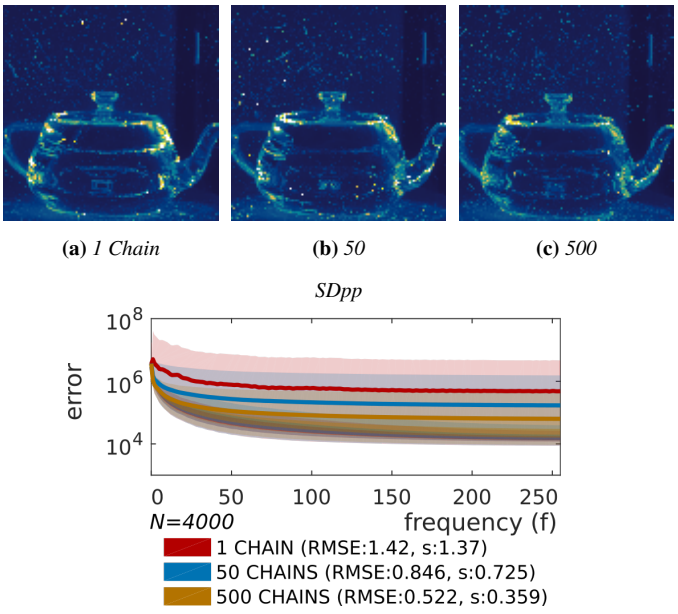
more technical aspects. One of the goals is to test whether breaking up MLT chains impacts performance. This is important because MLT relies on correlated chains of samples, which eventually cover the whole image. Our technique, based on short renderings, partitions these chains into independent runs.

We also provide more details about the link between a low-frequency peak and inter-pixel correlation.

Figures 7a and b show the scenes used in this section. Rendering budget scaling using time, as described in Equation 13, was used only for Section 5.1. In the subsequent sections we tested different configurations of the same rendering algorithm with the same number of samples, and hence wanted to eliminate unnecessary distortions of the results. We show example short renderings and standard deviation per pixel images (SDpp) where they provide additional insight over ESE. The computation time for computing ESE and SDpp is in the range of half an hour per scene.

5.1. Box Scene

Figure 8 shows changes of the ESE of a simple box scene (Figure 7a) when the size of the light source is varied, while keeping its power constant. We can see that the error of all algorithms grows when the light source becomes smaller. Path tracing is more sensitive than the MLT methods when looking at RMSE. The standard deviation of RMSE, i.e., a measure for outliers, on the other hand, increases more in MLT methods. The same is also visible when



(a) *1 Chain* **(b)** *50* **(c)** *500*

(d) *Error Spectrum Ensemble*

Figure 9: *SDpp and ESEs for various chain lengths in the door scene (Figure 7c). All strategies use the same total chain sample count of 50 times the pixel count. The 1 chain strategy contains most outliers but also the lowest ESE head (d). The SDpp (a - c) shows that much of the error is in a region with complicated refractions.*

looking at ESE. ESE also shows, that the MLT outliers consist of larger patches and not isolated pixels.

5.2. Impact of the Proxy Algorithm

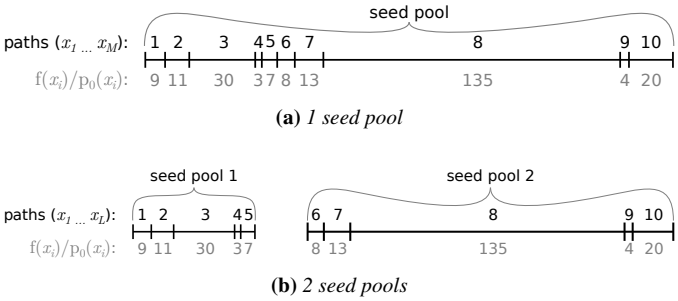
The behavior of the original algorithm and its proxy version are not necessarily identical. While this is the case for many MC algorithms (i.e., the result computed with the original algorithm would be equal to the result from the proxy if the same random numbers were used), it is not the case for MLT algorithms. In the original paper [VG97], it was proposed to maintain a single chain throughout the rendering process (though the possibility of using multiple chains was mentioned). Modern computer architectures made it beneficial to run several shorter chains in parallel and it is even required by our proxy algorithm. For these reasons we tested the impact of changing chain length and pooling strategies on four test scenes. The paper shows only a selection of them, see the supplemental material for the rest.

Chain Length. We tested the impact of breaking up Markov chains by computing ESEs and SDpp for 3 different chain strategies:

- 1 chain with $50 \times M$ (pixel count, 512^2) samples,
- 50 chains with M samples, and
- 500 chains with $0.1 \times M$ samples each.

The results show that the strategy with the longest chain was never the best. In 2 out of the four scenes, it contained significantly more outliers, an example is shown in Figure 9. We believe that this is due to the bi-directional mutation, which is often unable to lift the chain out of high-valued regions. The shorter chains are simply restarted, which limits the amount of time that a chain can be stuck in-place.

The bathroom scene (Figure 1a), which does not suffer from MLT outliers, also does not show significant differences between the chain lengths. This is intuitive considering that samples of a Markov chain are virtually independent if the distance between them is large enough, and an MLT chain should mix quickly enough in order to reach all parts of the image within a reasonable sample budget. In the short renderings, the chain is typically broken up only after it covered every pixel several times on average. Therefore, it does not matter whether the samples are completely independent by break-up or not.



(a) *1 seed pool*

(b) *2 seed pools*

Figure 10: *Conventionally, one seed pool is used in MLT, from which C chains are sampled proportionally to $f(x_i)/p_0(x_i)(a)$. It is possible to divide the L seed paths into S pools (b) and sample C/S starting paths from each.*

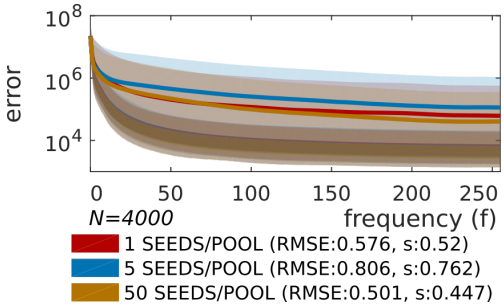
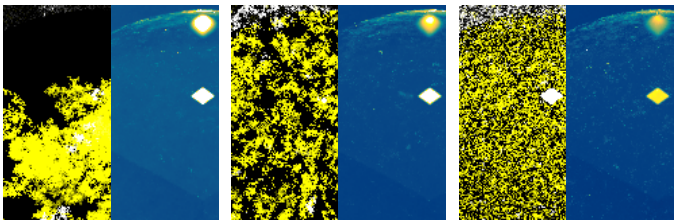


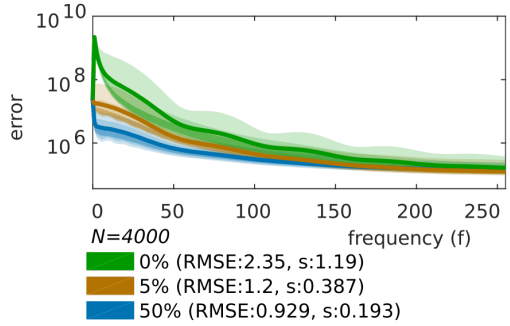
Figure 11: *ESEs for various seed pooling strategies in the torus scene (Figure 7b). The total number of luminance samples is 10k, which is partitioned randomly into 50, 10 and 1 seed pools with 1, 5 and 50 chains, each. The body and heads of the variants are almost equal, differences in tails, ensemble mean and RMSE are minor, indicating that the strategy plays only a minor role given large enough pools.*

Seed Pools. The strategy employed by Mitsuba [Jak10] is to sample all of the chains from a single seed pool (Figure 10a). It is not equivalent to the proxy algorithm, having at least one seed pool per short rendering (Figure 10b for $N = 2$). In the following, we



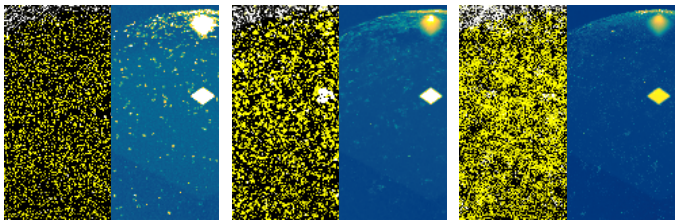
(a) *Large mutations 0%* **(b)** *5%* **(c)** *50%*

Example short renderings on the left and SDpp on the right



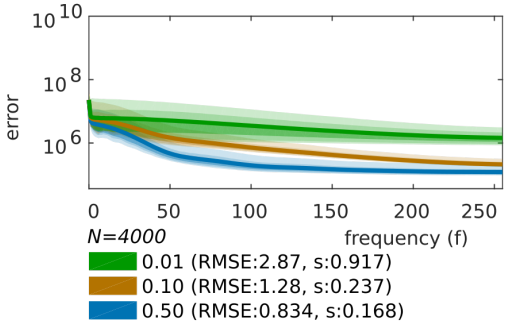
(d) *Error Spectrum Ensemble*

Figure 12: *Changing percentage of large mutations in primary sample space MLT (PSSMLT) in the box scene (Figure 7a): Large mutations generate independent path proposals, reducing correlation between pixels. In the renderings (a-c, left), long chains of correlated proposals show up as high-intensity patches. At 0%, global proposals only occur due to the sequence of short renderings performed by the proxy algorithm. SDpp shows that raising the percentage reduces the error in specular light effects, while ESE shows that there is less error in low frequencies.*



(a) *Mutation size 0.01* **(b)** *0.10* **(c)** *0.50*

Example short renderings on the left and SDpp on the right



(d) *Error Spectrum Ensemble*

Figure 13: *Changing small mutation size in PSSMLT in the box scene (Figure 7a): This parameter influences how much a path is modified between large jumps. With very small mutations the correlated chain segments barely move away from their initial pixel position (a, left), creating a very wide low frequency peak (d). Larger path modifications cause the correlated segments to move quickly into neighboring pixels, color patches have a larger radius (b and c, left), creating a narrower peak in low frequencies (d). The width of the low frequency peak decreases as the radius of correlation increases, similarly to the Fourier transform of a Gaussian kernel.*

describe the impact of the changed strategy. The MLT implementation was modified to include an additional parameter: The number of seeds per pool. Setting it equal to the number of chains (a parameter already present in our rendering system) results in Mitsuba's default strategy of using a single seed pool. Setting it to 1 will divide the original seed pool into one separate sub-pool per chain. This creates an algorithm which is equivalent to our proxy with N chains and one chain per pool. Finally, we compute the ESE using the standard procedure, a short-rendering budget of 50 samples per pixel, 50 chains, and

- a single pool (the original algorithm),
- 10 pools with (an in-between with 5 chains per pool), and
- 50 pools with (the equivalent with 1 chain per pool).

The data for the torus scene (Figure 7b) is shown in Figure 11. The results show that head and body are almost exactly the same. We deem differences in the tail, ensemble mean and RMSE inconclusive due to the scarcity of outliers.

In summary, the analysis in this section provides preliminary evidence that switching from a rendering algorithm to the corresponding proxy variant can affect its numerical performance – specifically, we found that performance of the proxy algorithm was generally superior. The chain length was particularly relevant, while the influence of the seed pooling strategy appears to be minor (see also the supplemental material).

5.3. Link Between Low-Frequency Peak and Inter-Pixel Correlation

In this section, we show how the typical low-frequency MLT peak reacts to changes in parameters that influence correlation. We use the primary sample space MLT (PSSMLT) algorithm [KSKAC02] and the scene shown in Figure 7a. PSSMLT is built on top of existing MC algorithms, for instance path tracing, and implements MCMC proposals by perturbing the current state, which consists of the "random numbers" that are supplied to the nested rendering algorithm. There are two types of mutations: Large ones propose independently sampled paths, and hence do not create correlation between pixels. In contrast, small mutations only slightly perturb the random number vector used to generate the path. Two main parameters control the behavior of these mutations: The percentage of large mutations, and the size (radius) of small mutations in the

random number domain. For this experiment, we changed one parameter and kept the other one steady.

Figure 12 shows results for a varying percentage of large mutations. It is not possible to reach true 0% with the proxy algorithm as every short render necessarily starts from an independent path and therefore acts as a 'large mutation'. We see that the peak is high when there are few large mutations and becomes smaller when the number increases. The radius, on the other hand, stays approximately the same. Setting the parameter to 100 % would result in an algorithm without correlation between pixels and a flat spectrum, with the only exception at the DC term.

The results for changing the size of small mutations are in Figure 13. Using a very small value of 0.01 causes the chain to stay very close to the starting pixel as shown in the example short renderings (Figure 13a), while higher values cause the change to propagate farther in image space (Figure 13c). This is also visible in the ESE, where the width of the low-frequency peak decreases as the mutation size increases. There are limits to this behavior, however.

6. Conclusion

We proposed a proxy rendering algorithm that encapsulates any unbiased method. It can be used to estimate standard deviation per-pixel images along with RMSE expectation and RMSE standard deviation. The proxy algorithm serves as the foundation of a tool for spectral and outlier analysis called error spectrum ensemble (ESE). It shows the existence and magnitude of outliers and the typical error levels where the short renderings fall, that are unaffected by outliers. We demonstrated the capabilities by changing parameters of known algorithms and examining the results. To ensure the reproducibility of our work, we provide a MATLAB implementation in the supplemental material.

Our method provides three error metrics that can be used to assess the convergence of rendering algorithms in ways that considerably extend simple quantities such as the overall MSE:

- Standard deviation per pixel images show the location and quantity of error, which is useful for identifying problematic lighting effects (e.g., Figures 9a through c).
- Individual short-time renderings produce information about the robustness of the convergence process, i.e., correlation, outliers etc. It is useful to examine them in order to gain insight into the qualitative behavior of the given algorithm (e.g., Figure 12a through 12c).
- Finally, ESE can be used to analyze the frequency content and outliers. It can be used to find suitable parameters, benchmarks, and to compare competing versions of algorithms (e.g., Section 5).

Limitations of the proposed tools are the relatively high computation cost, the restriction to unbiased algorithms with a convergence rate of $\Theta(1/\sqrt{N})$, and the relatively high complexity, compared to traditional scalar error metrics like MSE.

In future work, the visualization of our approach could be improved. A method for visualizing local pixel correlation might also be useful. The procedure for measuring the convergence rate of non-$\Theta(1/\sqrt{N})$ algorithms in Section 4.3 should be improved. In particular, a method that is more economical with computational resources and a better visualization are needed. The proposed method can not be computed on-line, which would make it possible to change rendering parameters on the fly. Another direction would be to couple the tools for measuring performance more closely to the algorithms, e.g., the mixing behavior of the Markov Chain in different parts of the scene.

Acknowledgements

We would like the anonymous reviewers for their valuable input. The "Door" scene was modeled after a scene by Eric Veach by Miika Aittala, Samuli Laine, and Jaakko Lehtinen. The "Sponza" scene is courtesy of Marko Dabrovic. The "Bookshelf", "Kitchen", "Bottle" and "Bathroom" scenes have been ported to Mitsuba by Tiziano Portenier. The "Torus" scene is based on a scene by Cline et al. This project was partly supported by the Academy of Finland, grant no. 277833 and by the EU MSCA-ITN project no. 813170. We acknowledge the computational resources provided by the Aalto "Science-IT" project.

Notation

$\mathrm{E}[X]$	Expectation of X
$\mathrm{Var}(X)$	Variance of X (pixel-wise for images)
$\langle I^n_m \rangle_K$	pixel m of image n of an array of short renderings with budget K (m and n are omitted in some contexts)
N, n	Number of short renderings, short rendering index
M, m	Number of pixels, pixel index
$\langle \mathcal{E} \rangle_N$	singed absolute error, i.e., $\langle I \rangle_N - I$, where I is the ground truth
$\langle \mathcal{E}^n \rangle_1$	estimated singed absolute error using the proxy algorithm, i.e., $\langle I^n \rangle_1 - \langle I \rangle_N$
$\widehat{\mathrm{MSE}}_x(\langle I \rangle_K)$	estimator of $\mathrm{E}[MSE(\langle I \rangle_K)]$ using x observations

References

[APSS01] ASHIKHMIN, MICHAEL, PREMOŽE, SIMON, SHIRLEY, PETER, and SMITS, BRIAN. "A variance analysis of the Metropolis light transport algorithm". *Computer & Graphics* 25.2 (2001), 287–294 2.

[ATS94] ARVO, JAMES, TORRANCE, KENNETH, and SMITS, BRIAN. "A framework for the analysis of error in global illumination algorithms". *Proceedings of SIGGRAPH '94*. 1994, 75–84 2.

[CTE05] CLINE, DAVID, TALBOT, JUSTIN, and EGBERT, PARRIS. "Energy redistribution path tracing". *ACM Transactions on Graphics (TOG)* 24.3 (2005), 1186 3.

[DHS*05] DURAND, FRÉDO, HOLZSCHUCH, NICOLAS, SOLER, CYRIL, et al. "A frequency analysis of light transport". *ACM Transactions on Graphics (TOG)* 24.3 (2005), 1115–1126 2.

[Jak10] JAKOB, WENZEL. "Mitsuba renderer". *http://www.mitsuba-renderer.org* (2010). URL: http://www.mitsuba-renderer.org 8.

[KMA*15] KETTUNEN, MARKUS, MANZI, MARCO, AITTALA, MIIKA, et al. "Gradient-domain path tracing". *ACM Transactions on Graphics (TOG)* 34.4 (2015), 123 2.

[KSKAC02] KELEMEN, CSABA, SZIRMAY-KALOS, LÁSZLÓ, ANTAL, GYÖRGY, and CSONKA, FERENC. "A simple and robust mutation strategy for the Metropolis light transport algorithm". *Computer Graphics Forum* 21.3 (2002), 531–540 6, 9.

[LD08] LAGAE, ARES and DUTRÉ, PHILIP. "A comparison of methods for generating poisson disk distributions". *Computer Graphics Forum* 27.1 (2008), 114–129 2.

[LKL*13] LEHTINEN, JAAKKO, KARRAS, TERO, LAINE, SAMULI, et al. "Gradient-domain metropolis light transport". *ACM Transactions on Graphics* 32.4 (2013), 95. ISSN: 07300301 2.

[PSC*15] PILLEBOUE, ADRIEN, SINGH, GURPRIT, COEURJOLLY, DAVID, et al. "Variance analysis for Monte Carlo integration". *ACM Transactions on Graphics (TOG)* 34.4 (2015), 124:1–124:14 2, 3, 7.

[RKZ12] ROUSSELLE, FABRICE, KNAUS, CLAUDE, and ZWICKER, MATTHIAS. "Adaptive rendering with non-local means filtering". *ACM Transactions on Graphics (TOG)* 31.6 (2012), 195 2.

[SK13] SUBR, KARTIC and KAUTZ, JAN. "Fourier analysis of stochastic sampling strategies for assessing bias and variance in integration". *ACM Transactions on Graphics (TOG)* 32.4 (2013), 128 2.

[SKDP99] SZIRMAY-KALOS, LASZLO, DORNBACH, PETER, and PURGATHOFER, WERNER. "On the start-up bias problem of metropolis sampling". *Winter School of Computer Graphics '99*. 1999, 273–280 2.

[VG97] VEACH, ERIC and GUIBAS, LEONIDAS J. "Metropolis light transport". *Proceedings of SIGGRAPH '97*. 1997, 65–76 2, 8.

[WJM17] WHITTLE, JOSS, JONES, MARK W., and MANTIUK, RAFAŁ. "Analysis of reported error in Monte Carlo rendered images". *Visual Computer* 33.6-8 (2017), 705–713 2, 7.

[ZAD15] ZIRR, TOBIAS, AMENT, MARCO, and DACHSBACHER, CARSTEN. "Visualization of coherent structures of light transport". *Computer Graphics Forum* 34.3 (2015), 491–500 2.