Practical Error Estimation for Denoised Monte Carlo Image Synthesis

Arthur Firmino
KeyShot, Technical University of Denmark
Denmark
arthur.firmino@keyshot.com

Ravi Ramamoorthi
University of California, San Diego
USA
ravir@cs.ucsd.edu

Jeppe Revall Frisvad
Technical University of Denmark
Denmark
jerf@dtu.dk

Henrik Wann Jensen
KeyShot
USA
henrik.jensen@keyshot.com

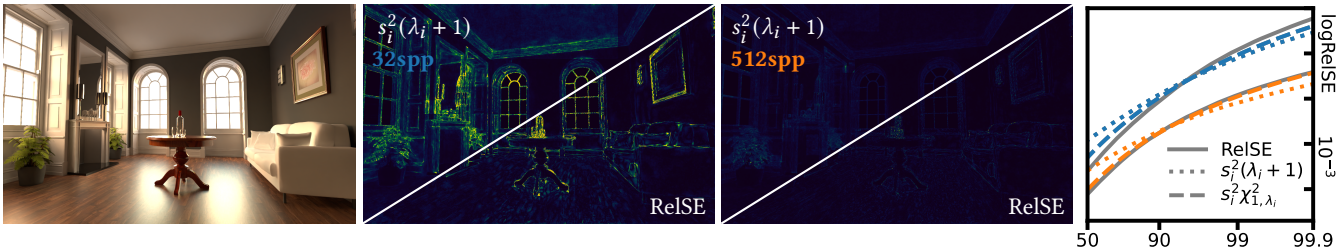

Figure 1: An example demonstrating that our error estimation framework for Monte Carlo denoised images reliably estimates an image's ground truth relative squared error (RelSE) distribution at different average sample counts (spp). We show that the RelSE of a denoised pixel i follows a scaled noncentral chi-squared distribution, with scale s_i^2 and noncentrality λ_i. To estimate these parameters, we hierarchically aggregate noisy per pixel estimates of error and variance. Knowing s_i^2 and λ_i, we can accurately estimate the image's error distribution, visualized per pixel (middle) and as the upper half of a logistic-logarithmic percentile plot (rightmost), leading to a robust stopping criterion for denoised Monte Carlo image synthesis.

ABSTRACT

We present a practical global error estimation technique for Monte Carlo ray tracing combined with deep learning based denoising. Our method uses aggregated estimates of bias and variance to determine the squared error distribution of the pixels. Unlike unbiased estimates for classical Monte Carlo ray tracing, this distribution follows a noncentral chi-squared distribution, under reasonable assumptions. Based on this, we develop a stopping criterion for denoised Monte Carlo image synthesis that terminates rendering once a user specified error threshold has been achieved. Our results demonstrate that our error estimate and stopping criterion work well on a variety of scenes, and that we are able to achieve a given error threshold without the user specifying the number of samples needed.

CCS CONCEPTS

• **Computing methodologies** → **Rendering**.

KEYWORDS

Error estimation, denoising, Monte Carlo, path tracing, stopping criterion.

SIGGRAPH Conference Papers '24, July 27-August 1, 2024, Denver, CO, USA

ACM ISBN 979-8-4007-0525-0/24/07.
https://doi.org/10.1145/3641519.3657511

ACM Reference Format:
Arthur Firmino, Ravi Ramamoorthi, Jeppe Revall Frisvad, and Henrik Wann Jensen. 2024. Practical Error Estimation for Denoised Monte Carlo Image Synthesis. In *Special Interest Group on Computer Graphics and Interactive Techniques Conference Conference Papers '24 (SIGGRAPH Conference Papers '24), July 27-August 1, 2024, Denver, CO, USA*. ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3641519.3657511

1 INTRODUCTION

Monte Carlo algorithms such as path tracing have become the de facto standard in production rendering [Keller et al. 2015; Christensen and Jarosz 2016; Pharr 2018], owing to their simplicity of implementation, ability to handle complex scenes and light transport scenarios, and scalability. Although initially impeded by high computational costs due to their large sample count requirement, advances in ray tracing specific hardware as well as image denoising algorithms have enabled their widespread adoption.

As sample variance between different scenes and across image space within the same scene is generally inhomogeneous, rendering to within an acceptable error threshold can often require some amount of trial-and-error work as the end user tries to ascertain a suitable sample count for a given scene. Monte Carlo denoising which effectively trades variance for bias further complicates this process as the easily perceived stochastic noise is no longer available to help guide the artist, and important details may be blurred out of the image entirely. While recent works have sought to improve the convergence of Monte Carlo denoising through adaptive sampling [Vogels et al. 2018; Salehi et al. 2022; Firmino et al. 2023], as well as ensuring consistency [Firmino et al. 2022; Back et al. 2022; Gu et al. 2022; Back et al. 2023], they do not consider the problem of

estimating the global error to enable a stopping criterion for Monte Carlo rendering of a denoised image.

When using an unbiased rendering technique, image error can simply be estimated using sample variance. This error estimate can then function as a stopping criterion, and a user can specify a desired maximum error instead of a sample count [Lee et al. 1985]. With the widespread use of image denoising, we need a different error estimation method as the denoiser invariably introduces bias (and an estimator's mean-squared error, MSE, is the sum of its variance and squared bias). Most denoisers today are based on deep learning and the bias they introduce is not obvious.

An error-predicting network [Vogels et al. 2018], a dual-buffering method [Back et al. 2022], and Stein's unbiased risk estimate (SURE) [Li et al. 2012; Firmino et al. 2022] have been used to estimate the MSE of denoised images. We focus on SURE and consider its suitability for guiding a stopping criterion. Some of the issues we tackle are that SURE is very noisy for low sample counts and SURE really estimates the expected value of per pixel squared error and not the error of a given realization. Based on our investigation, we present an improved error estimation method practical as a guide for stopping of Monte Carlo denoised images. Figure 1 provides an example of our estimated error in comparison to the ground truth error at 32 and 512 samples per pixel (spp) on average.

2 RELATED WORK

A stopping criterion often goes hand-in-hand with adaptive sampling because an estimate of the distribution of error in a rendered image can serve both purposes. In unbiased rendering, an estimate of the variance is an estimate of the squared error. For n samples in a pixel, the sum of squared differences between sample and sample mean is expected to follow a chi-squared distribution with $n-1$ degrees of freedom. An option is then a per-pixel stopping criterion that changes with the number of samples based on the maximum tolerated variance and the cumulative distribution function (cdf) of the chi-squared distribution [Lee et al. 1985]. In a different approach, Dippé and Wold [1985] expect an error inversely proportional to the square root of the number of samples and estimate the factor of proportionality using rendered pixel values for several different sample counts. This factor is used to estimate the number of samples required in a pixel to end up with a pixel error below a desired bound. Alternatively, the width of a confidence interval for the pixel value can be used as a stopping criterion [Purgathofer 1987; Tamstorf and Jensen 1997]. Such early work did not consider the error of denoised images, but we use the observation that the squared error follows a chi-squared distribution [Lee et al. 1985].

Adaptive hierarchical integration is a way to improve path tracing convergence by controlling the sampling rate in nodes of a kd tree [Kajiya 1986]. One approach is to refine the kd tree according to a variance estimate and the area of each node and then use a confidence interval and a coverage condition as the stopping criterion [Painter and Sloan 1989]. With a focus on directional discontinuities in the rendered image (edge-like features), Guo [1998] uses progressive refinement of image blocks to locate the parts of an image that require more samples. For every progressive update, each of the few selected pixels in each block is rendered using the hierarchical integration of Kajiya [1986] with the stopping criterion of Painter and Sloan [1989]. We take inspiration from this early work and use a kd tree to adaptively split the image into blocks that we use for averaging of noisy per-pixel SURE estimates.

Mitchell [1987] uses a contrast threshold to have a per-pixel stopping criterion more closely related to visual perception of error. The contrast of a pixel is computed from different sample values obtained in a supersampling cell. Other work has applied a perceptually based threshold model too [Bolin and Meyer 1998; Ramasubramanian et al. 1999; Myszkowski 2002]. We incorporate spatial contrast sensitivity in our method with inspiration from the ꟻlip error metric by Andersson et al. [2020].

Various stopping criteria are based on different pixel quality metrics. One approach is to use entropy from information theory and keep refining the image until the samples in a region provide sufficiently homogeneous information [Rigau et al. 2003a,b; Xu et al. 2007]. Another metric is based on fuzziness from fuzzy set theory [Xu et al. 2006]. Some researchers train a discriminator to learn binary human noise classification (noisy or not) and use it as a stopping criterion with no user parameter [Constantin et al. 2016; Takouachet et al. 2017; Buisine et al. 2021a,b]. These approaches however do not perform error estimation, so we cannot use them to render an image to be approximately within some error bound.

To have a robust stopping criterion, biased rendering techniques need an error estimation framework different from the variance-based approach used for unbiased techniques [Overbeck et al. 2009; Hachisuka et al. 2010]. Denoising introduces bias and its application means that another error estimation framework is needed regardless of the technique used for rendering the input to the denoiser. Such error estimation and associated stopping criteria have been derived for non-neural denoisers [Moon et al. 2013; Kalantari and Sen 2013]. In the case of neural denoisers, probabilistic error bounds are not as easily derived.

Because the neural denoiser is more of a black box, we need more general error estimation techniques. These are either expensive to evaluate or difficult to bound. One approach is per-pixel error estimation based on the difference between two images rendered with equal sample count (dual-buffering) [Dammertz et al. 2010; Rousselle et al. 2012]. Dammertz et al. [2010] combined this approach with a hierarchical refinement of image blocks similar to ours. Back et al. [2022] used the dual-buffer approach together with a neural denoiser, but this requires rendering and denoising of two images for each iteration. An error-predicting network [Vogels et al. 2018] depends on its training dataset and may be arbitrarily off, making it hard to bound even probabilistically. Use of SURE is another general error estimation technique [Li et al. 2012; Rousselle et al. 2013] that has been used with neural denoising [Firmino et al. 2022, 2023]. The usefulness of SURE for estimating the global image error in the context of neural denoising of Monte Carlo rendered images is however unknown. Thus, we use hierarchical image block refinement to improve SURE-based error estimation for neural denoising and develop a stopping criterion for denoised Monte Carlo rendering based on this error estimation.

3 ERROR ESTIMATION FRAMEWORK

Our proposed framework is based on estimating the squared error distribution over a Monte Carlo denoised image. In Section 3.1, we

lay out our theoretical assumptions and show that the squared error of a denoised pixel follows a noncentral chi-squared distribution. In Section 3.2, we present SURE and extend it to estimate relative squared error and a modified error metric that incorporates spatial contrast sensitivity. This is followed by Section 3.3 in which we detail how to compute SURE using the Jacobian-vector product, and Section 3.4 where we present an adaptive algorithm for block averaging the noisy error estimate. In Section 3.5, we show how these estimates relate to the parameters of the aforementioned noncentral chi-squared distribution, and finally in Section 3.6 we present our stopping criterion that is based on computing the upper percentile of an image's error distribution. An overview of our error estimation framework and the steps involved is later detailed in Algorithm 2.

3.1 Theory

Suppose $\mathcal{F} : \mathbb{R}^N \mapsto \mathbb{R}^N$ is a denoiser and $X \in \mathbb{R}^N$ is a non-denoised rendered image of N pixels times channels. A denoised Monte Carlo pixel of index i is denoted $\mathcal{F}_i(X)$, and to understand its error distribution, we can approximate it by a weighted average of neighbouring pixels plus bias:

$$\mathcal{F}_i(X) = W_i^\top X + b_i \tag{1}$$

with $W_i \in \mathbb{R}^N$ and $b_i \in \mathbb{R}$. This approximation equals the first two terms of the Taylor series of $\mathcal{F}_i(X)$. Given that the denoiser's neural network is at least piece-wise smooth, this approximation should be well founded for small enough input deviations. Assuming X is normally distributed with mean $\mu \in \mathbb{R}^N$ and covariance matrix $\Sigma \in \mathbb{R}^{N\times N}$, the random variable $\mathcal{F}_i(X) - \mu_i$ representing the error of $\mathcal{F}_i(X)$ is then also normally distributed with mean and variance, respectively,

$$\mu_{e,i} = W_i^T \mu + b_i - \mu_i\,, \quad s_{e,i}^2 = W_i^T \Sigma W_i\,. \tag{2}$$

We base our assumption of the distribution of X on the central limit theorem, which states that in limit of many samples the distribution of the sample mean approaches a normal distribution. Following from these assumptions, the square of $\mathcal{F}_i(X) - \mu_i$ is then distributed according to a scaled noncentral chi-squared distribution with one degree of freedom [Muirhead 2009], $s_i^2 \chi^2_{1,\lambda_i}$, with scale and noncentrality, respectively,

$$s_i^2 = s_{e,i}^2\,, \quad \lambda_i = \mu_{e,i}^2 / s_{e,i}^2\,. \tag{3}$$

It is important to clarify the distinction between the measured squared error $(\mathcal{F}_i(X) - \mu_i)^2$, often reported in research results, and its expectation value $\mathbb{E}[(\mathcal{F}_i(X) - \mu_i)^2]$. The latter is a constant while the former follows the aforementioned distribution which is positively skewed. This positive skewness makes it impractical to apply a maximum bound on the squared error, even for denoised images, unless that bound is probabilistic. Figure 2 exemplifies this distinction. These conclusions regarding the squared error distribution also extend to relative squared error (RelSE), which is simply squared error divided by a constant:

$$\text{RelSE} = \frac{(\mathcal{F}_i(X) - \mu_i)^2}{\mu_i^2 + 10^{-2}}\,. \tag{4}$$

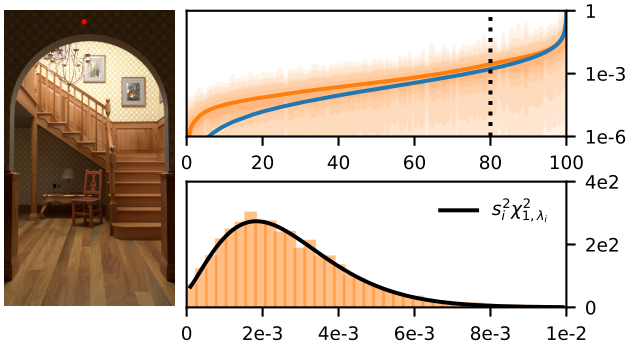

Figure 2: Percentile plot of the relative squared error (RelSE) of a Monte Carlo denoised image (top) and of its distribution for an individual pixel (bottom). In the top plot, the blue line shows sorted RelSE, where X is rendered with 32 spp. The orange line shows the expected value of RelSE, computed by rendering X 4096 times with distinct seeds, and the shaded regions illustrate different quantiles of the pixel's RelSE distribution. In the bottom plot, we show a normalized histogram of the RelSE of a pixel at the 80th percentile, and the probability density function of a noncentral chi-squared distribution with parameters $s_i^2 = 2.52 \times 10^{-4}$ and $\lambda_i = 9.42$ computed from the 4096 images.

Figure 3: Visualizing SURE (middle) on a positive/negative color scale. Note the high variance of its per-pixel estimates, often leading to negative estimates despite the non-negativity of the actual RelSE (right). This high variance motivates the hierarchical aggregating of these per-pixel estimates in our error estimation framework. The estimates are from a 32 spp rendered and denoised image (left).

3.2 Error Estimation using SURE

Stein's unbiased risk estimate (SURE) can be used for estimating the MSE of a point estimate of the mean of normally distributed random variables in an unbiased manner [Stein 1981]. When used for estimating the expected value of per-pixel squared errors in denoised images [Li et al. 2012], its unbiased-ness rests in part on the assumption that the sample means of the rendered image follow a normal distribution, an assumption we explore further below. It is relevant to state that SURE estimates the expected squared error,

$$\mathbb{E}[\text{SURE}[\mathcal{F}_i(X)]] = \mathbb{E}[(\mathcal{F}_i(X) - \mu_i)^2]\,, \tag{5}$$

and not the squared error of a given realization, $(\mathcal{F}_i(X) - \mu_i)^2$. Using SURE, the estimate for the per-pixel expected squared error of a denoised image is given by [Liu 1994]

$$\text{SURE}[\mathcal{F}_i(X)] = (\mathcal{F}_i(X) - X_i)^2 + 2(J_{\mathcal{F}}(X)\,\Sigma)_{ii} - \Sigma_{ii}, \tag{6}$$

where Σ is the covariance matrix of X, which may be estimated during rendering, and $J_{\mathcal{F}}(X)$ is the denoiser's Jacobian matrix at X. The middle term, $2(J_{\mathcal{F}}(X)\,\Sigma)_{ii}$, can be estimated using either the Monte Carlo SURE method [Ramani et al. 2008; Firmino et al. 2022], or using the Jacobian-vector product [Firmino et al. 2023].

The error estimate introduced above suffers from high variance, as illustrated in Figure 3, made further challenging by the fact that we are often interested in the error of denoised images at relatively low sample counts compared to non-denoised images. This is especially true when denoising with neural denoisers.

While SURE presents us with a method of estimating the expected squared error, this metric skews heavily to the brightest parts of the image. We can improve upon this simply by dividing by the square of the maximum channel of the denoised pixel, plus some positive epsilon, yielding a biased estimate of the relative squared error:

$$\mathrm{SURE}^{\mathrm{r}}[\mathcal{F}_i(X)] = \frac{\mathrm{SURE}[\mathcal{F}_i(X)]}{\max_{j\in\mathcal{P}(i)} \mathcal{F}_j(X)^2 + 10^{-2}}, \tag{7}$$

where $\mathcal{P}(i)$ denotes the pixel of index i, such that $\max_{j\in\mathcal{P}(i)} \mathcal{F}_j(X)$ is the maximum value of that pixel's three color channels. The bias stems from the fact that in general $\mathbb{E}[1/X] \neq 1/\mathbb{E}[X]$, and from the dependence between the two estimates in the ratio. Since the variance of $\mathcal{F}_i(X)$ is small relative to that of the numerator, we expect this additional bias to be acceptable for our purposes.

Qualitative inspection of error images reveals that relative squared error often peaks on single pixels near small aliased details such as sharp edges, which may not be perceptually significant under regular viewing conditions. Inspired by the ꟻLIP error metric [Andersson et al. 2020], we can augment the error estimate by incorporating spatial contrast sensitivity, expressed as a series of low-pass filters in the Yc_xc_z opponent space, with the width and shape of these filters being determined by the human eye's sensitivity to spatial changes in the Y, c_x, and c_z channels. We rely on the implementation by Andersson et al. [2020] to define these filters. As the operations involved, color transformations and filter convolutions, are linear, we can derive an unbiased error estimate that incorporates these. Concretely, given a matrix $M \in \mathbb{R}^{N\times N}$, we find that

$$\mathrm{SURE}_M[\mathcal{F}_i(X)] = (M\mathcal{F}(X) - MX)_i^2 + (M\Sigma(2J_{\mathcal{F}}^{\top}(X) - I)M^{\top})_{ii}\,, \tag{8}$$

such that $\mathbb{E}[\mathrm{SURE}_M[\mathcal{F}_i(X)]] = \mathbb{E}[(M\mathcal{F}(X) - M\mu)_i^2]$, with notation as in Eq. 6 and I as the identity matrix. The proof follows from the substitution of X with $\mu + \epsilon$, where $\epsilon \sim \mathcal{N}(0, \Sigma)$, and from Stein's Lemma. Denoting M_i as the i-th row of M, then

$$\begin{aligned}(M_i\mathcal{F}(X) - M_iX)^2 &= (M_i\mathcal{F}(X) - M_i\mu - M_i\epsilon)^2 \\ &= (M_i\mathcal{F}(X) - M_i\mu)^2 - 2(M_i\epsilon)(M_i\mathcal{F}(X) - M_i\mu) + (M_i\epsilon)^2. \end{aligned}\tag{9}$$

We expand the middle term to calculate its expected value as follows

$$\mathbb{E}[(M_i\epsilon)M_i(\mathcal{F}(X) - \mu)] = \mathbb{E}\left[\sum_{j=1}^{N} M_{ij}\left(\sum_{k=1}^{N} M_{ik}\epsilon_j(\mathcal{F}_k(X) - \mu_k)\right)\right]$$

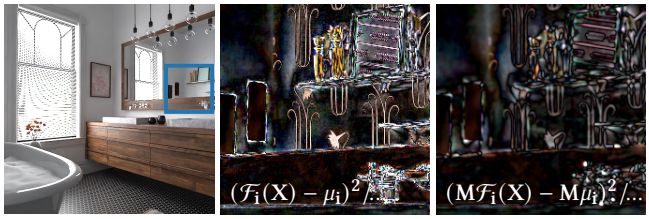

Figure 4: Comparing relative squared error with (right) and without (middle) incorporating spatial contrast sensitivity, as described in Sec. 3.2. The modified metric places less emphasis on error that is not perceptually relevant under regular viewing conditions, such as along edges and fine details that are not easily perceived in the denoised image (left).

$$\begin{aligned}&= \sum_{j=1}^{N} M_{ij}\left(\sum_{k=1}^{N} M_{ik}\mathbb{E}[\epsilon_j\mathcal{F}_k(X)]\right) \\ &= \sum_{j=1}^{N} M_{ij}\left(\sum_{k=1}^{N} M_{ik}\left(\sum_{l=1}^{N}\Sigma_{jl}\mathbb{E}\left[\frac{\mathcal{F}_k}{\partial x_l}(X)\right]\right)\right) \\ &= \sum_{j=1}^{N} M_{ij}\left(\sum_{k=1}^{N} M_{ik}(\mathbb{E}[J_{\mathcal{F}}(X)]\Sigma)_{kj}\right) = \sum_{j=1}^{N} M_{ij}\left(M\,\mathbb{E}[J_{\mathcal{F}}(X)]\Sigma\right)_{ji}^{\top} \\ &= \left(M(M\,\mathbb{E}[J_{\mathcal{F}}(X)]\Sigma)^{\top}\right)_{ii} = (M\Sigma\,\mathbb{E}[J_{\mathcal{F}}^{\top}(X)]M^{\top})_{ii}\end{aligned}$$

The third equality involves use of the multivariate form of Stein's Lemma [Liu 1994], and a similar procedure can be used to show that $\mathbb{E}[(M_i\epsilon)^2] = (M\Sigma M^{\top})_{ii}$. The proof concludes by applying the expected value operator to Eq. 9 and using the derived solutions, then rearranging it in the form of the right-hand side of Eq. 8, thus showing it to be an estimate of the expected perceptually augmented squared error. The relative version of this estimate is then

$$\mathrm{SURE}^{\mathrm{r}}_M[\mathcal{F}_i(X)] = \frac{\mathrm{SURE}_M[\mathcal{F}_i(X)]}{\max_{j\in\mathcal{P}(i)}(M\mathcal{F}(X))_j^2 + 10^{-2}}\,. \tag{10}$$

Figure 4 illustrates the difference in error magnitudes when incorporating spatial contrast sensitivity, expressed in the matrix M. We later report results with and without this addition.

3.3 Computing SURE

To estimate the terms

$$2(J_{\mathcal{F}}(X)\,\Sigma)_{ii} \quad \text{and} \quad (M\Sigma(2J_{\mathcal{F}}^{\top}(X) - I)M^{\top})_{ii}$$

of Eqs. 6 and 8, respectively, we derive the quadratic form whose expectation matches the desired quantity. It is known, given a square matrix $\Lambda \in \mathbb{R}^{N\times N}$ and a centered random-variable ϵ with covariance matrix Σ, that $\mathbb{E}[\epsilon^T\Lambda\epsilon] = \mathrm{tr}[\Lambda\Sigma]$ [Muirhead 2009]. It can also be shown that $\mathbb{E}[\epsilon_i(\Lambda\epsilon)_i] = (\Lambda\Sigma)_{ii}$. From this, we arrive at the estimates

$$\mathbb{E}[\epsilon_i(J_{\mathcal{F}}(X)\epsilon)_i] = (J_{\mathcal{F}}(X)\,\Sigma)_{ii} \tag{11}$$

$$\mathbb{E}[(M\epsilon)_i(M(2J_{\mathcal{F}}(X) - I)\epsilon)_i] = (M\Sigma(2J_{\mathcal{F}}^{\top}(X) - I)M^{\top})_{ii}\,. \tag{12}$$

The random variable ϵ is sampled from $\mathcal{N}(0, \hat{\Sigma})$, where $\hat{\Sigma}$ is the estimated covariance matrix of X. The matrix $\hat{\Sigma}$ is in practice diagonal except for elements representing pairs of color channels from the

same pixel. However, since $\hat{\Sigma}$ may not be positive semi-definite, we use singular value decomposition to generate ϵ. Computing $J_F(X)v$ for $v \in \mathbb{R}^N$ can be done using the Jacobian-vector product (also known as forward-mode auto-differentiation) [Baydin et al. 2018], and Mv is implicitly calculated by applying the corresponding operations to v.

3.4 Block Averaging of SURE

Individual per-pixel error estimates using SURE, although unbiased, are unreliable due to their high variance as previously illustrated. We propose averaging neighbouring error estimates by means of a kd tree over image space, which adaptively splits blocks of pixels along the block's longest axis as long as the block-averaged SURE estimates are expected to remain reliable.

In the homoskedastic case ($\Sigma = \sigma^2 I$), results from Bellec and Zhang [2021] imply $\mathrm{SURE}[\mathcal{F}(X)]$ is within a small fraction of the actual error $\|\mu - \mathcal{F}(X)\|^2$ when the actual error is of an order greater than $\sigma^2 N^{1/2}$. For clarity, we rewrite the stated inequality as:

$$N^{-1/2} \sum_{i \in [1..N]} \sigma^2 \ll \sum_{i \in [1..N]} \mathrm{SURE}[F_i(X)].$$

We base the splitting criterion of our kd tree on this result and split a given block $\mathcal{B}$ if the following condition would hold true for both of its children blocks, $\mathcal{B}_1$ and $\mathcal{B}_2$, and their intersection with each of the color channels $C_k, k \in \{1, 2, 3\}$:

$$|\mathcal{B}_j \cap C_k|^{-1/2} \sum_{i \in \mathcal{B}_j \cap C_k} \hat{\Sigma}^r_{ii} < \sum_{i \in \mathcal{B}_j \cap C_k} \mathrm{SURE}^{\mathrm{r}}[\mathcal{F}_i(X)], \qquad (13)$$

where $\hat{\Sigma}^r_{ii} = \hat{\Sigma}_{ii} / (\max_{j \in \mathcal{P}(i)} \mathcal{F}_j(X)^2 + 10^{-2})$. In the case of using $\mathrm{SURE}^{\mathrm{r}}_M$ then an estimate of $(M\hat{\Sigma}M^\top)_{ii}$ is used in place of $\hat{\Sigma}_{ii}$. Figure 5 exemplifies the result of recursively applying this splitting criterion, which is detailed in Algorithm 1.

ALGORITHM 1: Adaptive block-wise splitting of image plane.

Input : Estimated relative covariance matrix $\hat{\Sigma}^{\mathrm{r}} \in \mathbb{R}^{N \times N}$ (sparse 3×3 block diagonal), and error estimates $\mathrm{SURE}^{\mathrm{r}} \in \mathbb{R}^N$.
Output: Disjoint set of blocks $\{\mathcal{B}_0, \mathcal{B}_1, \ldots\}$ spanning the image.

```
ComputeBlocks(Σ̂^r, SURE^r, B ← Block(Width, Height, 3))
if LongestAxisLength(B) > 1 then
    splitBlock ← true
    B_1, B_2 ← SplitAlongLongestAxis(B)
    foreach color channel k ∈ {1, 2, 3} do
        a ← Sum(Σ̂^r, B_1∩C_k)/√|B_1 ∩ C_k| < Sum(SURE^r, B_1∩C_k)
        b ← Sum(Σ̂^r, B_2∩C_k)/√|B_2 ∩ C_k| < Sum(SURE^r, B_2∩C_k)
        splitBlock ← splitBlock and a and b
    if splitBlock then
        B_1 ← ComputeBlocks(Σ̂^r, SURE^r, B_1)
        B_2 ← ComputeBlocks(Σ̂^r, SURE^r, B_2)
        return {B_1, B_2}
return {B}
```

3.5 Estimating Error Distribution Parameters

While the adaptive kd tree is useful for finding suitable minimum block sizes for which the mean SURE value is reliable, this can still result in an overly coarse error estimate over image space due to the large block size, especially at lower sample counts, which do not capture the heterogeneous error distribution within each block. To estimate values relating to the image's error distribution, while capturing its heterogeneity, we propose modelling the error distribution of each pixel i as a scaled noncentral chi-squared distribution with one degree of freedom, $s_i^2 \chi^2_{1,\lambda_i}$. We set the scale parameter s_i^2 equal to the denoised pixel's estimated relative variance $\mathrm{Var}^{\mathrm{r}}[\mathcal{F}_i(X)]$, and set the noncentrality parameter such that the error's mean, $\mathbb{E}[s_i^2 \chi^2_{1,\lambda_i}] = s_i^2(\lambda_i + 1)$, is equal to the block-estimated SURE value scaled in proportion to the block's per-pixel variance estimates:

$$\lambda_i = \max\left(\frac{\sum_{j \in \mathcal{B}(i) \cap C(i)} \mathrm{SURE}^{\mathrm{r}}[\mathcal{F}_j(X)]}{\sum_{j \in \mathcal{B}(i) \cap C(i)} \mathrm{Var}^{\mathrm{r}}[\mathcal{F}_j(X)] + 10^{-6}}, 1 \right) - 1 \qquad (14)$$

where $\mathcal{B}(i)$ and $C(i)$ denote the block and color channel of index i, respectively. Figure 6 visualizes the estimated parameters and their relation to the expected relative squared error.

Figure 5: Block averaged SURE and false color visualization of the blocks, for two different sample counts, and for which the block sizes were computed by recursively applying the splitting criterion of Eq. 13, as in Algorithm 1 (magnitude of SURE exaggerated for clarity).

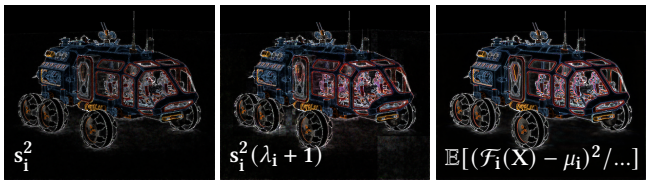

Figure 6: Estimated parameters of the per-pixel error distribution $s_i^2 \chi^2_{1,\lambda_i}$ (magnitudes exaggerated for clarity), from a 32 spp rendering. The scale parameter s_i^2 (left) is the denoised pixel's estimated relative variance, while $s_i^2(\lambda_i + 1)$ (middle), with λ_i calculated as in Eq. 14, corresponds to the error distribution's mean. The rightmost images, which the middle image should ideally match, shows the expected RelSE and was computed from 20 independent image.

3.6 SURE-Based Stopping Criterion

We assume rendering is performed progressively, with additional samples being rendered with each progressive update and that these samples may be distributed uniformly or according to some adaptive sampling scheme. We also assume that the denoiser $\mathcal{F}$ has either negligible or vanishing bias with increasing sample count, such that achieving the desired quality is possible. This is achieved

using a consistent denoiser [Back et al. 2023] or denoising with post-correction [Firmino et al. 2022; Gu et al. 2022].

To propose a robust and practical implementation of a stopping criterion for denoised Monte Carlo rendering, so as to automatically terminate rendering when some measure of quality is achieved, with this measure being consistent across scenes, we formalize such a criterion as any predicate function befitting the following form:

$$P_{\mathcal{F}}(X|\tau) : \mathbb{R}^N \mapsto \{0, 1\}$$

where τ is a user parameter, returning 1 when rendering should terminate. To evaluate the performance of such a stopping function, it is useful if there exists a ground truth stopping function, $P^{gt}_{\mathcal{F}}(X|\tau, \mu)$, having knowledge of the reference image $\mu = \mathbb{E}[X]$. Given such a function, we wish to find a practical method for computing $P_{\mathcal{F}}(X|\tau)$ that minimizes its discrepancy with the ground truth.

Given parameters s_i^2 and λ_i, the cumulative distribution function $F_i(x)$ of the proposed error distribution $s_i^2 \chi^2_{1,\lambda_i}$ of pixel i can be computed, as can the cumulative distribution function of the error distribution of the entire image,

$$F(x) = (1/N) \sum_i^N F_i(x). \tag{15}$$

Given an error threshold τ, we can then estimate the proportion of pixels whose error is below that threshold which forms the basis of our proposed stopping function:

$$P_{\mathcal{F}}(X|p, \tau) = \mathbb{1}\left[F(\tau) \geq p\right]. \tag{16}$$

In practice, we fix the parameter $p \in [0, 1]$ to 0.999, which would correspond to 99.9% of the pixels having error less than the user specified threshold τ. Computation of the ground truth stopping function, for evaluation purposes, involves merely comparing the error at the given percentile to the threshold. An overview of our complete error estimation framework is in Algorithm 2.

ALGORITHM 2: Overview of our error estimation framework.

Input : Current iteration's noisy rendered estimates $X \in \mathbb{R}^N$, estimates of pixel channel covariance matrix $\hat{\Sigma} \in \mathbb{R}^{N \times N}$ (sparse 3×3 block diagonal), and error threshold τ.

Output : Result of our SURE-based stopping criterion.

```
// Denoise and Compute SURE (Section 3.3)
ϵ            ← SampleNormal(0, Σ̂)
F(X), (Jϵ)   ← EvaluateDenoiserAndJVP(X, ϵ)
SURE         ← (F(X) − X) ⊙ (F(X) − X) + ϵ ⊙ (2(Jϵ) − ϵ)
SURE^r       ← SURE          / (PixelWiseMax(F(X))^2 + 10^−2)
Var^r        ← (Jϵ) ⊙ (Jϵ) / (PixelWiseMax(F(X))^2 + 10^−2)
Σ̂^r          ← Σ̂             / (PixelWiseMax(F(X))^2 + 10^−2)
// Block splitting (Section 3.4, Algorithm 1)
{B_0, B_1, ...} ← ComputeBlocks(Σ̂^r, SURE^r)
// Estimating distribution parameters (Section 3.5)
foreach index i ∈ [1..N] do
    C_k ← ColorChannelOf(i)
    B   ← BlockOf(i, {B_0, B_1, ...})
    λ_i ← max( Sum(SURE^r, B∩C_k) / (Sum(Var^r, B∩C_k) + 10^−6), 1 ) − 1
    s_i^2 ← Var^r_i
// Compute SURE-based stopping function (Section 3.6)
F ← 0
foreach index i ∈ [1..N] do
    F ← F + ComputeNoncentralChiSquaredCDF(τ/s_i^2, λ_i)/N
return F ≥ 99.9%
```

4 IMPLEMENTATION

In our experiments, we used Mitsuba 3 [Jakob et al. 2022] CPU renderer to render images and Intel's Open Image Denoise [Áfra 2019] in combination with a post-correction denoising method [Firmino et al. 2022] to denoise images and to avoid the otherwise non-vanishing bias from preventing stopping. Iterative adaptive sampling, as described by Firmino et al. [2023], was also used in all of our experiments to achieve faster error convergence over uniform sampling. Our stopping criterion was also evaluated iteratively, with an iteration step size of 32 (average) samples per pixel. This is reflected in our terminal sample counts being multiples of 32. Denoising and computation of the pixel-wise error estimates (Section 3.3) was performed on the GPU using PyTorch [Paszke et al. 2019], while the block averaging of SURE and computation of the distribution parameters and stopping function (Sections 3.4, 3.5, and 3.6) was performed on the CPU. We tested our method on a set of 20 publicly available scenes [Bitterli 2016]. For a representative scene (BATHROOM) at 1024×1024 resolution, denoising and computation of SURE took 90 milliseconds, block averaging 29 milliseconds, and computation of the stopping function another 11 milliseconds. Compared to the mean per-iteration rendering duration of 3.2 seconds, the relative overhead is only 4%. When compared to the adaptive sampling of Firmino et al. [2023], the inclusion of our stopping criterion incurs a runtime penalty of just 1-2%, see Table 2, because denoising and the Jacobian-vector product are computed for the adaptive sampling in any case.

5 RESULTS AND DISCUSSION

Our error estimation framework estimates parameters of the underlying error distributions to evaluate the mixed cumulative distribution function at the specified threshold. We evaluate the proposed stopping criterion by comparing it to its ground truth version, which computes the RelSE using the reference image and finds the percentage of pixels below the specified threshold. In all our results, we fix the percentile value to 99.9%, such that quality can be tuned by only one parameter. Results are shown in Table 1 for our experiments using SURE^{r} and SURE^{r}_M. We use smaller error thresholds for the SURE^{r}_M experiments as the magnitude of the estimates is generally smaller. In our results, we deviate from the original definition of Eq. 4 for relative square error, replacing its denominator with that of Eq. 7 (or Eq. 10), so as to match our estimates.

Our experiments show good agreement, here defined as stopping within one third or one step (32 samples per pixel) of the true sample count, in 52 out of 60 cases (87%), and in 49 out of 60 cases (82%) for the SURE^{r} and SURE^{r}_M experiments, respectively. Failure cases generally appear to be scene-dependent, spanning both experiments and different error thresholds. We examined one of these cases in Figure 10, finding that scene elements giving rise to extreme sample variance lead to slow convergence and erroneous error estimates. Pixels with such sample distributions would require more samples for their means to converge to a normal distribution (never

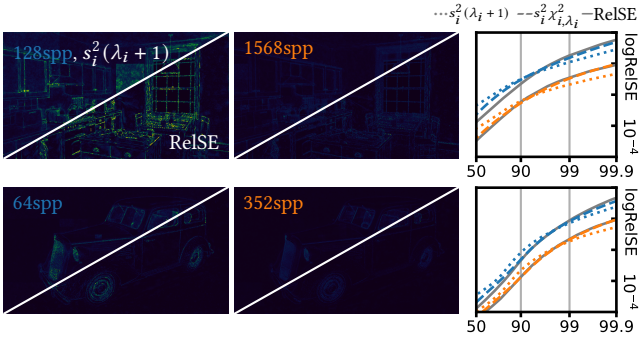

Figure 7: Here we visualize, in false color, the estimated expected RelSE (based on the computed parameters s_i^2 and λ_i; upper left cuts) and the actual RelSE (lower right cuts), for two different scenes (KITCHEN and CAR) and at two different stopping thresholds (0.050 and 0.010). The percentile error plots (rightmost), show the predicted error from the distribution of $s_i^2\chi^2_{1,\lambda_i}$ (computed by inverting the CDF of Eq. 15), and the actual error (RelSE). The estimated expected RelSE $s_i^2(\lambda_i+1)$ is also plotted (its line computed by sorting individual estimates), which is lower than the actual error at the upper percentiles as predicted.

converging if variance is infinite), breaking one of the assumptions underlying SURE.

We compare some terminal images for different error thresholds in Figure 8, and in Figure 7 we visualize the relative squared error images and show how our framework accurately estimates different percentiles of the images' relative squared error distribution, similar to how is shown in Figure 1.

Our framework accounts not only for variance but also for the bias in denoised images, as a consequence of relying on SURE which estimates both quantities. In Figure 9 we compare our error estimation to previous work which only estimates variance [Firmino et al. 2023] and consequently underestimates squared error. From this comparison we conclude that our method and its block averaging of SURE, reliably estimates the non-centrality parameter λ_i (Eq. 14), and thus accounts for bias when estimating squared error.

The practical use case of our proposed framework and its associated stopping criterion is in enabling the automatic termination of Monte Carlo denoised rendering at an appropriate sample count, an otherwise time consuming trial-and-error process for the user.

6 LIMITATIONS AND FUTURE WORK

Sources of Bias in Error Estimates. Applying SURE to Monte Carlo denoising requires assuming the sample means are normally distributed, else the estimate may be biased. Provided sample variance is finite and samples independent, the sample mean's distribution tends to a normal distribution in the limit of many samples, as per the central limit theorem. How many samples are required, for the assumption to be valid, cannot be answered in general and depends on the sample distributions, which in practice are diverse [Elek et al. 2019], possibly plagued by outliers [Zirr et al. 2018], or having infinite variance [Kalos 1963; Georgiev et al. 2013].

Top-Down Block Size Determination. The splitting criteria defined in Eq. 13 will fail to split if the estimated error from SURE is erroneously small or negative. This may happen due to the aforementioned bias or due to variance inherent in the estimates despite aggregating. For this reason, better error estimation results are sometimes had when pre-splitting to smaller blocks (e.g. 200×200 pixel sized blocks) before applying the criterion. The result by Bellec and Zhang [2021] which inspires our criterion, does not account for the extra variance from our estimation of the covariance matrix Σ and its heteroskedasticity.

Perceptually Relevant Error. We incorporated spatial contrast sensitivity into the squared error metric used in this work to increase its perceptual relevance, similar to Andersson et al. [2020] in the construction ꟻLIP. Due to our requirement that these additions involve only linear operations, and being limited to squared differences, we did not incorporate other perceptual aspects into the metric, and differences were computed in the linear RGB color space with each pixel's channels being weighted equally. Finding a more perceptually relevant metric which may still be estimated unbiasedly by Eq. 8 remains future work.

7 CONCLUSION

We have presented a framework for estimating the error of Monte Carlo denoised images. Noting that each pixel's squared error should follow a noncentral chi-squared distribution, we estimate the distribution's parameters by aggregating estimates of bias and variance by way of a kd tree. Given these parameters, we can estimate the error distribution of the denoised image, including the very top percentiles, which we use to guide our stopping criterion. In our experiments, we find close agreement between our terminal sample counts and the sample count at which the specified error threshold is actually achieved. Our error estimation framework thus provides a reliable method of finding a suitable sample count for a given desired quality, without the need for trial-and-error by the end user.

ACKNOWLEDGMENTS

This research is a part of PRIME which is funded by the European Union's Horizon 2020 research and innovation programme under the Marie Skłodowska Curie grant agreement No. 956585. This work was supported in part by NSF grant 2212085. We also acknowledge gifts from Adobe, Google and Qualcomm, and the Ronald L. Graham Chair.

REFERENCES

Attila T. Áfra. 2019. Intel® Open Image Denoise. https://www.openimagedenoise.org/.

Pontus Andersson, Jim Nilsson, Tomas Akenine-Möller, Magnus Oskarsson, Kalle Åström, and Mark D. Fairchild. 2020. FLIP: A Difference Evaluator for Alternating Images. *Proc. ACM Comput. Graph. Interact. Tech.* 3, 2 (2020), 15–1. https://doi.org/10.1145/3406183

Jonghee Back, Binh-Son Hua, Toshiya Hachisuka, and Bochang Moon. 2022. Self-supervised post-correction for Monte Carlo denoising. In *SIGGRAPH 2022 Conference Papers*. ACM, 18:1–18:8. https://doi.org/10.1145/3528233.3530730

Jonghee Back, Binh-Son Hua, Toshiya Hachisuka, and Bochang Moon. 2023. Input-dependent uncorrelated weighting for Monte Carlo denoising. In *SIGGRAPH Asia 2023 Conference Papers*. ACM, 9:1–9:10. https://doi.org/10.1145/3610548.3618177

Atılım Güneş Baydin, Barak A. Pearlmutter, Alexey Andreyevich Radul, and Jeffrey Mark Siskind. 2018. Automatic differentiation in machine learning: a survey. *Journal of Marchine Learning Research* 18, 153 (2018), 1–43. http://jmlr.org/papers/v18/17-468.html

Pierre C Bellec and Cun-Hui Zhang. 2021. Second-order Stein: SURE for SURE and other applications in high-dimensional inference. *The Annals of Statistics* 49, 4 (2021), 1864–1903.

Benedikt Bitterli. 2016. Rendering Resources. https://benedikt-bitterli.me/resources/.

Mark R. Bolin and Gary W. Meyer. 1998. A perceptually based adaptive sampling algorithm. In *SIGGRAPH '98*. 299–309. https://doi.org/10.1145/280814.280924

Jérôme Buisine, André Bigand, Rémi Synave, Samuel Delepoulle, and Christophe Renaud. 2021a. Stopping criterion during rendering of computer-generated images based on SVD-entropy. *Entropy* 23, 1 (2021), 75. https://doi.org/10.3390/e23010075

Jérôme Buisine, Fabien Teytaud, Samuel Delepoulle, and Christophe Renaud. 2021b. Guided-generative network for noise detection in Monte-Carlo rendering. In *International Conference on Machine Learning and Applications (ICMLA)*. IEEE, 61–66. https://doi.org/10.1109/ICMLA52953.2021.00018

Per H. Christensen and Wojciech Jarosz. 2016. The path to path-traced movies. *Foundations and Trends in Computer Graphics and Vision* 10, 2 (2016), 103–175. https://doi.org/10.1561/0600000073

Joseph Constantin, Ibtissam Constantin, Andre Bigand, and Denis Hamad. 2016. Perception of noise in global illumination based on inductive learning. In *International Joint Conference on Neural Networks (IJCNN)*. IEEE, 5021–5028. https://doi.org/10.1109/IJCNN.2016.7727861

Holger Dammertz, Johannes Hanika, Alexander Keller, and Hendrik Lensch. 2010. A hierarchical automatic stopping condition for Monte Carlo global illumination. In *Proceedings of the Winter School of Computer Graphics*. 159–164.

Mark A. Z. Dippé and Erling Henry Wold. 1985. Antialiasing through stochastic sampling. *Computer Graphics (SIGGRAPH '85)* 19, 3 (July 1985), 69–78. https://doi.org/10.1145/325334.325182

Oskar Elek, Manu M. Thomas, and Angus Forbes. 2019. Learning patterns in sample distributions for Monte Carlo variance reduction. arXiv:1906.00124 [cs.GR]. https://doi.org/10.48550/arXiv.1906.00124

Arthur Firmino, Jeppe Revall Frisvad, and Henrik Wann Jensen. 2022. Progressive denoising of Monte Carlo rendered images. *Computer Graphics Forum* 41, 2 (May 2022), 1–11. https://doi.org/10.1111/cgf.14454

Arthur Firmino, Jeppe Revall Frisvad, and Henrik Wann Jensen. 2023. Denoising-aware adaptive sampling for Monte Carlo ray tracing. In *SIGGRAPH 2023 Conference Proceedings*. ACM, 32:1–32:11. https://doi.org/10.1145/3588432.3591537

Iliyan Georgiev, Jaroslav Krivanek, Toshiya Hachisuka, Derek Nowrouzezahrai, and Wojciech Jarosz. 2013. Joint importance sampling of low-order volumetric scattering. *ACM Transactions on Graphics* 32, 6 (2013), 164:1–164:14. https://doi.org/10.1145/2508363.2508411

Jeongmin Gu, Jose A. Iglesias-Guitian, and Bochang Moon. 2022. Neural James-Stein combiner for unbiased and biased renderings. *ACM Transactions on Graphics* 41, 6 (December 2022), 262:1–262:14. https://doi.org/10.1145/3550454.3555496

Baining Guo. 1998. Progressive radiance evaluation using directional coherence maps. In *SIGGRAPH '98*. ACM, 255–266. https://doi.org/10.1145/280814.280888

Toshiya Hachisuka, Wojciech Jarosz, and Henrik Wann Jensen. 2010. A progressive error estimation framework for photon density estimation. *ACM Transactions on Graphics* 29, 6 (2010), 144:1–144:12. https://doi.org/10.1145/1882261.1866170

Wenzel Jakob, Sébastien Speierer, Nicolas Roussel, Merlin Nimier-David, Delio Vicini, Tizian Zeltner, Baptiste Nicolet, Miguel Crespo, Vincent Leroy, and Ziyi Zhang. 2022. *Mitsuba 3 Renderer*. https://mitsuba-renderer.org

James T. Kajiya. 1986. The rendering equation. *Computer Graphics (SIGGRAPH '86)* 20, 4 (August 1986), 143–150. https://doi.org/10.1145/15922.15902

Nima Khademi Kalantari and Pradeep Sen. 2013. Removing the noise in Monte Carlo rendering with general image denoising algorithms. 32, 2pt1 (2013), 93–102. https://doi.org/10.1111/cgf.12029

Malvin H. Kalos. 1963. On the estimation of flux at a point by Monte Carlo. *Nuclear science and engineering* 16, 1 (1963), 111–117.

Alexander Keller, Luca Fascione, Marcos Fajardo, Iliyan Georgiev, Per Christensen, Johannes Hanika, Christian Eisenacher, and Gregory Nichols. 2015. The path tracing revolution in the movie industry. In *SIGGRAPH 2015 Courses*. ACM. https://doi.org/10.1145/2776880.2792699

Mark E. Lee, Richard A. Redner, and Samuel P. Uselton. 1985. Statistically optimized sampling for distributed ray tracing. *Computer Graphics (SIGGRAPH '85)* 19, 3 (July 1985), 61–68. https://doi.org/10.1145/325334.325179

Tzu-Mao Li, Yu-Ting Wu, and Yung-Yu Chuang. 2012. SURE-based optimization for adaptive sampling and reconstruction. *ACM Transactions on Graphics* 31, 6 (November 2012), 194:1–194:9. https://doi.org/10.1145/2366145.2366213

Jun S Liu. 1994. Siegel's formula via Stein's identities. *Statistics & Probability Letters* 21, 3 (1994), 247–251. https://doi.org/10.1016/0167-7152(94)90121-X

Don P. Mitchell. 1987. Generating antialiased images at low sampling densities. *Computer Graphics (SIGGRAPH '87)* 21, 4 (July 1987), 65–72. https://doi.org/10.1145/37401.37410

Bochang Moon, Jong Yun Jun, JongHyeob Lee, Kunho Kim, Toshiya Hachisuka, and Sung-Eui Yoon. 2013. Robust image denoising using a virtual flash image for Monte Carlo ray tracing. *Computer Graphics Forum* 32, 1 (2013), 139–151. https://doi.org/10.1111/cgf.12004

Robb J. Muirhead. 2009. *Aspects of multivariate statistical theory*. John Wiley & Sons.

Karol Myszkowski. 2002. Perception-driven global illumination and rendering computation. In *Advanced Computer Systems (ACS 2001)*. Springer, 267–288. https://doi.org/10.1007/978-1-4419-8530-9_22

Ryan S. Overbeck, Craig Donner, and Ravi Ramamoorthi. 2009. Adaptive wavelet rendering. *ACM Transactions on Graphics* 28, 5 (2009), 140:1–140:12. https://doi.org/10.1145/1618452.1618486

James Painter and Kenneth Sloan. 1989. Antialiased ray tracing by adaptive progressive refinement. *Computer Graphics (SIGGRAPH '89)* 23, 3 (July 1989), 281–288. https://doi.org/10.1145/74333.74362

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Yang, Zach DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. PyTorch: an imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems (NeurIPS 2019, Vol. 32)*.

Matt Pharr. 2018. Guest editor's introduction: special issue on production rendering. *ACM Transactions on Graphics* 37, 3 (2018), 28:1–28:4. https://doi.org/10.1145/3212511

Werner Purgathofer. 1987. A statistical method for adaptive stochastic sampling. *Computers & Graphics* 11, 2 (1987), 157–162. https://doi.org/10.1016/0097-8493(87)90029-X

Sathish Ramani, Thierry Blu, and Michael Unser. 2008. Monte-Carlo SURE: a black-box optimization of regularization parameters for general denoising algorithms. *IEEE Transactions on Image Processing* 17, 9 (September 2008), 1540–1554. https://doi.org/10.1109/TIP.2008.2001404

Mahesh Ramasubramanian, Sumanta N. Pattanaik, and Donald P. Greenberg. 1999. A perceptually based physical error metric for realistic image synthesis. In *SIGGRAPH '99*. ACM, 73–82. https://doi.org/10.1145/311535.311543

Jaume Rigau, Miquel Feixas, and Mateu Sbert. 2003a. Entropy-based adaptive sampling. In *Graphics Interface (GI 2003)*. Canadian Human-Computer Communications Society and A K Peters, 149–158. https://doi.org/10.20380/GI2003.18

Jaume Rigau, Miquel Feixas, and Mateu Sbert. 2003b. Refinement criteria based on f-divergences. In *Rendering Techniques (EGWR 2003)*. Eurographics Association, 260–269. https://doi.org/10.2312/EGWR/EGWR03/260-269

Fabrice Rousselle, Claude Knaus, and Matthias Zwicker. 2012. Adaptive rendering with non-local means filtering. *ACM Transactions on Graphics* 31, 6 (November 2012), 195:1–195:11. https://doi.org/10.1145/2366145.2366214

Fabrice Rousselle, Marco Manzi, and Matthias Zwicker. 2013. Robust denoising using feature and color information. *Computer Graphics Forum* 32, 7 (2013), 121–130. https://doi.org/10.1111/cgf.12219

Farnood Salehi, Marco Manzi, Gerhard Roethlin, Romann Weber, Christopher Schroers, and Marios Papas. 2022. Deep adaptive sampling and reconstruction using analytic distributions. *ACM Transactions on Graphics* 41, 6 (2022), 259:1–259:16. https://doi.org/10.1145/3550454.3555515

Charles M. Stein. 1981. Estimation of the mean of a multivariate normal distribution. *The Annals of Statistics* 9, 6 (November 1981), 1135–1151.

Nawel Takouachet, Samuel Delepoulle, Christophe Renaud, Nesrine Zoghlami, and João Manuel R. S. Tavares. 2017. Perception of noise and global illumination: toward an automatic stopping criterion based on SVM. *Computers & Graphics* 69 (2017), 49–58. https://doi.org/10.1016/j.cag.2017.09.008

Rasmus Tamstorf and Henrik Wann Jensen. 1997. Adaptive sampling and bias estimation in path tracing. In *Rendering Techniques '97 (EGWR)*. Springer, 285–295. https://doi.org/10.1007/978-3-7091-6858-5_26

Thijs Vogels, Fabrice Rousselle, Brian McWilliams, Gerhard Röthlin, Alex Harvill, David Adler, Mark Meyer, and Jan Novák. 2018. Denoising with kernel prediction and asymmetric loss functions. *ACM Transactions on Graphics* 37, 4 (August 2018), 124:1–124:15. https://doi.org/10.1145/3197517.3201388

Qing Xu, Mateu Sbert, Miquel Feixas, and Jizhou Sun. 2007. A new adaptive sampling technique for Monte Carlo global illumination. In *Computer-Aided Design and Computer Graphics (CAD/Graphics)*. IEEE, 191–196. https://doi.org/10.1109/CADCG.2007.4407879

Qing Xu, Lianping Xing, Wei Wang, and Mateu Sbert. 2006. Adaptive sampling based on fuzzy inference. In *GRAPHITE 2006*. ACM, 311–317. https://doi.org/10.1145/1174429.1174482

Tobias Zirr, Johannes Hanika, and Carsten Dachsbacher. 2018. Re-weighting firefly samples for improved finite-sample Monte Carlo estimates. *Computer Graphics Forum* 37, 6 (2018), 410–421. https://doi.org/10.1111/cgf.13335

Table 1: To evaluate our proposed stopping criterion, we compare its terminal sample count to the sample count at which the given threshold τ is actually achieved by 99.9% of pixels (values in parentheses), for twenty different scenes and six different thresholds. For the top three rows, the threshold is specified as relative squared error, and for the bottom three rows it is as perceptual relative squared error which incorporates spatial contrast sensitivity (denoted by the subscript M). The cell colors indicate if our criterion stopped too early (in red) or too late (in blue), in terms relative to the parenthesized values and with colors determined by the scale (left). Light red or blue indicate that the terminal sample count from our stopping criterion closely agrees with ground truth.

τ	BATHROOM	BATHROOM2	BEDROOM	CAR	CAR2	CLASSROOM	COFFEE	DINING-ROOM	GLASS-OF-WATER	HOUSE	KITCHEN	LAMP	LIVING-ROOM	LIVING-ROOM-2	LIVING-ROOM-3	ROVER	SPACESHIP	STAIRCASE	STAIRCASE2	TEAPOT-FULL
0.050	1280 (1536)	96 (64)	352 (448)	64 (64)	64 (64)	544 (512)	64 (64)	64 (64)	2624 (2784)	96 (64)	128 (160)	64 (64)	288 (288)	64 (96)	384 (192)	64 (64)	96 (64)	64 (64)	32 (64)	896 (1408)
0.020	4672 (4864)	320 (416)	1408 (1664)	128 (160)	64 (64)	1824 (2272)	64 (96)	160 (96)	7808 (8288)	128 (128)	640 (576)	64 (64)	992 (992)	256 (288)	416 (992)	96 (96)	96 (96)	96 (96)	64 (128)	4192 (7776)
0.010	11776 (13344)	1472 (1696)	4032 (4576)	352 (352)	64 (64)	6368 (7200)	192 (224)	160 (192)	17280 (21600)	256 (224)	1568 (1440)	160 (160)	2528 (2624)	704 (736)	896 (4288)	192 (160)	160 (192)	192 (224)	320 (384)	14112 (–)
0.010_M	608 (608)	64 (96)	352 (384)	64 (64)	64 (64)	416 (672)	64 (64)	64 (64)	1024 (992)	64 (64)	96 (160)	64 (64)	192 (192)	64 (64)	224 (736)	64 (64)	64 (64)	64 (64)	64 (64)	480 (704)
0.005_M	1536 (1568)	128 (288)	960 (992)	64 (64)	64 (64)	896 (1600)	64 (64)	128 (64)	2240 (2048)	64 (64)	352 (352)	64 (64)	448 (480)	160 (192)	256 (1952)	64 (64)	96 (96)	64 (64)	96 (128)	1376 (1760)
0.003_M	2944 (2976)	288 (672)	1888 (1856)	96 (96)	64 (64)	1952 (3104)	96 (96)	160 (128)	4032 (3616)	96 (96)	704 (704)	96 (64)	896 (896)	352 (352)	640 (4544)	96 (64)	128 (160)	64 (96)	192 (224)	2368 (5824)

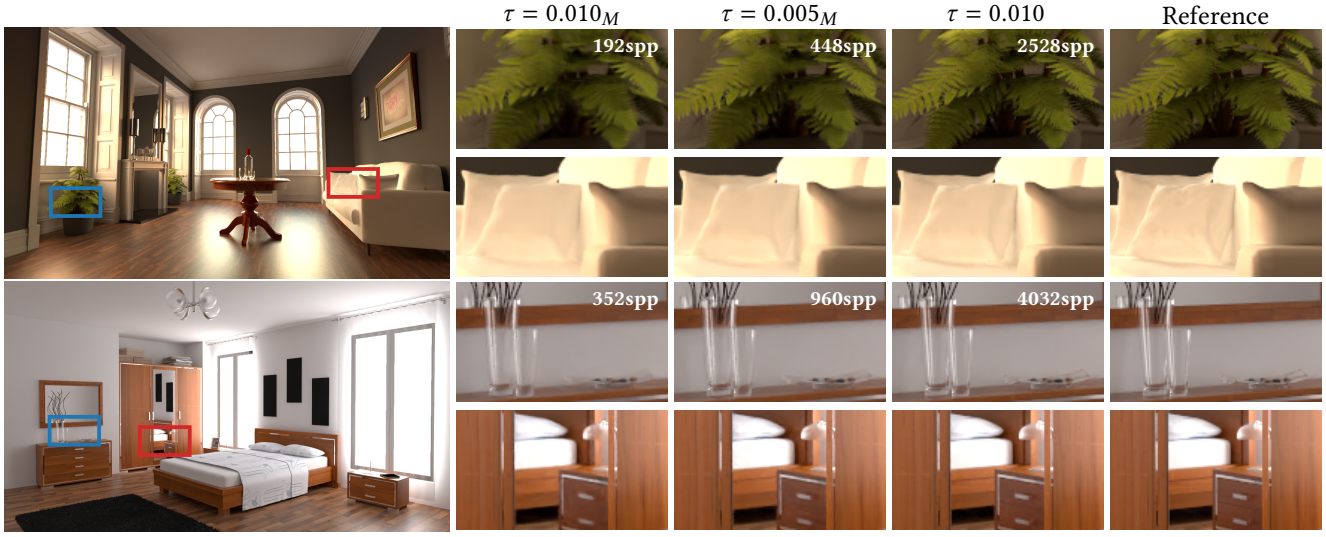

Figure 8: Comparing terminal images for different error thresholds, we note their general differences in terms of perceived quality. Differences are notable, for example in the blurring of foliage details (top row), between the 0.010_M and 0.010 error thresholds, the former's metric incorporating spatial contrast sensitivity while the latter does not. While this blurring is noted in close ups of the image denoised to the lowest of the perceptual thresholds, this is expected as those details are not easily perceived under regular viewing conditions of the whole image, and therefore not as penalized by the metric. Scenes are LIVING-ROOM and BEDROOM for the top and bottom rows respectively.

Table 2: Equal quality (8×32spp iterations) comparison of total runtime for our error estimation framework (top values) against the adaptive sampling method of Firmino et al. [2023] (values in parentheses), for the 20 different scenes listed in Table 1. In this scenario, the inclusion of our proposed stopping criterion incurs a small overhead of only 1-2% as denoising and computation of the Jacobian-vector product are already performed.

BATHROOM	BATHROOM2	BEDROOM	CAR	CAR2	CLASSROOM	COFFEE	DINING-ROOM	GLASS-OF-WATER	HOUSE	KITCHEN	LAMP	LIVING-ROOM	LIVING-ROOM-2	LIVING-ROOM-3	ROVER	SPACESHIP	STAIRCASE	STAIRCASE2	TEAPOT-FULL
33.36s	28.65s	30.56s	21.68s	18.81s	31.92s	15.57s	32.90s	31.25s	19.87s	28.35s	20.46s	33.29s	29.08s	37.75s	15.30s	22.06s	31.26s	35.20s	24.22s
(32.94s)	(28.28s)	(30.20s)	(21.32s)	(18.45s)	(31.56s)	(15.26s)	(32.54s)	(30.88s)	(19.51s)	(27.98s)	(20.05s)	(32.93s)	(28.72s)	(37.39s)	(15.05s)	(21.70s)	(30.89s)	(34.79s)	(23.86s)

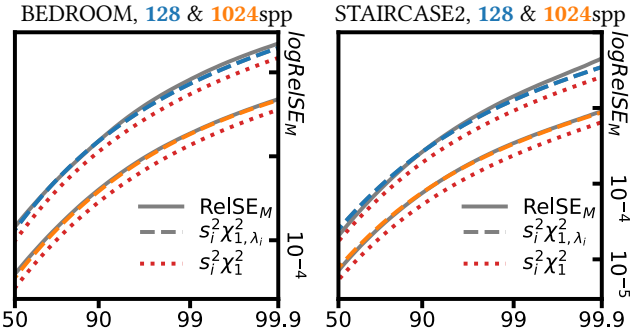

Figure 9: We compare our SURE-based error estimation against an error estimate that ignores the denoiser's bias, relying solely on estimates of its variance [Firmino et al. 2023]. We do this as though its squared error were chi-squared distributed ($s_i^2\chi_1^2$), rather than according to the noncentral chi-squared distribution ($s_i^2\chi^2_{1,\lambda_i}$) for which our SURE-based framework estimates the noncentrality parameter λ_i. The method based on previous work, shown in red in the above percentile error plots, clearly underestimates the error, indicating that an error estimation framework for denoised images should account for bias.

τ	Orig.	Mod.
0.050	384 (192)	32 (32)
0.020	416 (992)	64 (64)
0.010	896 (4288)	160 (224)
0.010_M	224 (736)	32 (64)
0.005_M	256 (1952)	64 (64)
0.003_M	640 (4544)	128 (128)

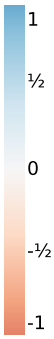

Figure 10: Examining one of the failure cases, LIVING-ROOM-3, we found excessive sample variance arising from volumetric scattering within the red vases, leading to inaccurate error estimates and early stopping. Replacing the culpable geometry's material with an opaque material leads to significantly better results. The table and scale follow the same definitions as for Table 1.