Converging Algorithm-Agnostic Denoising for Monte Carlo Rendering

ELENA DENISOVA, University of Florence, Italy
LEONARDO BOCCHI, University of Florence, Italy

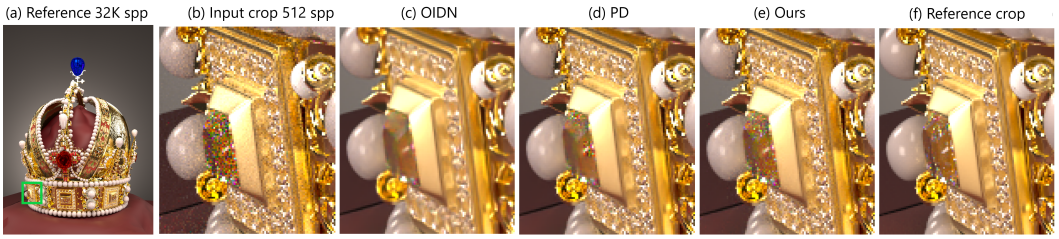

Fig. 1. Our method (e) addresses the issue of excessive smoothing observed in denoising of Monte Carlo estimates (b), even when using high-quality denoising algorithms such as *Intel® Open Image Denoise* (OIDN) with albedo and normal auxiliary features (c). By leveraging convergence curve prediction, our approach ensures numerical convergence independently of the denoising algorithm, eliminating the need for denoising at every iteration. Our method yields results comparable to those achieved with learning-based *progressive denoising* (PD) and, in some cases, even outperforms it, as demonstrated in (d).

Denoising Monte Carlo rendered images is a critical challenge in computer graphics, traditionally addressed in post-processing phases. Recent advances have shifted focus towards integrating denoising directly into progressive rendering. This approach not only blends denoised outputs with noisy inputs to enhance visual quality but also guides adaptive sampling through variance estimation of denoised outputs. In this paper, we introduce a novel method that predicts the blending weight of denoised images directly during progressive rendering. Our technique adjusts the blending weight for each pixel based on error estimates, effectively 'skipping' certain iterations of Monte Carlo Path Tracing (MCPT) and mimicking adaptive sampling *a posteriori*. A key innovation of our approach is an analytical method that ensures the blended output converges accurately to the reference image. Our method does not rely on deep-learning techniques, making it immediately applicable to any denoising algorithm. We demonstrate that our method enhances visual quality and allows for blending between noisy and denoised images, even those obtained at different MCPT iterations. This not only streamlines the rendering process but also improves efficiency and output fidelity.

CCS Concepts: • **Computing methodologies** → **Ray tracing**.

Additional Key Words and Phrases: Monte Carlo Path Tracing, Converging Denoising, Analytical Blending

ACM Reference Format:
Elena Denisova and Leonardo Bocchi. 2024. Converging Algorithm-Agnostic Denoising for Monte Carlo Rendering. *Proc. ACM Comput. Graph. Interact. Tech.* 7, 3, Article 40 (July 2024), 16 pages. https://doi.org/10.1145/3675384

Authors' Contact Information: Elena Denisova, elena.denisova@unifi.it, University of Florence, Florence, FI, Italy; Leonardo Bocchi, leonardo.bocchi@unifi.it, University of Florence, Florence, FI, Italy.

ACM 2577-6193/2024/7-ART40
https://doi.org/10.1145/3675384

1 Introduction

In 1986, Kajija [Kajiya 1986] introduced Monte Carlo Path Tracing (MCPT), a rendering technique capable of generating images that closely resemble photographs when accurately modeling both lighting and material properties. However, the inherent stochastic nature of Monte Carlo (MC) simulations inevitably introduces noise, particularly in initial iterations and areas with insufficient lighting, which stands as a primary drawback of this method [Cook 1986]. As scenes and lighting configurations grow more complex, the number of iterations required to produce a clear, noise-free image increases accordingly [Shirley and Morley 2008]. Employing noise reduction as a post-processing step on the partially converged image can significantly reduce the time needed to generate an acceptable, though not always perfectly accurate, result [Firmino et al. 2022]. In the field of denoising, a common challenge arises when managing high-frequency elements within rendered images. Denoising algorithms often struggle to differentiate these details from noise, leading to unintended over-blurring even at very high sample counts. This loss of fine detail is evident in the spectral analysis, where the lower amplitude of the higher frequencies indicates a suppression of these crucial components within the denoised image (see Figure 1 of Supplementary).

However, preserving the details and ensuring that the denoised image accurately converges to the reference can be crucial, particularly in fields like biomedical rendering, where three dimensional representation has demonstrated its utility in quickly comprehending traumas in anatomically complex areas, aiding in surgical planning, simulation, and training [Bueno et al. 2021; Ebert et al. 2017]. A recent advancement in this realm is the introduction of the Advanced Realistic Rendering Technique (AR^2T), inspired by MCPT and tailored for biomedical volumes [Denisova et al. 2023].

Recent studies advocate for blending denoised outputs with noisy inputs to enhance results [Gu et al. 2022; Yang 2024] and ensure convergence to the ground truth [Firmino et al. 2022]. However, these approaches rely on neural networks trained on denoised images, necessitating re-training whenever the denoising algorithm changes.

In our work, we propose an analytical solution for blending unbiased (hypothetically noisy MC estimate) and biased (denoised) images, independent of the denoising algorithm employed, whether traditional or deep learning-based. Our method accommodates both progressive denoising, applied iteratively after each MC iteration to refine accumulated estimates, and progressive rendering, where denoising ceases after a predetermined number of iterations. In alternative, our method can be used as a final step in the MCPT algorithm, helping to reconstruct details potentially lost during denoising when the algorithm converges. In contrast to real-time rendering with minimal ray budgets [Donnelly et al. 2024; Schied et al. 2017], our method is specifically designed for progressive rendering, a method commonly used for generating photorealistic images in the movie industry [Bako et al. 2017; Christensen et al. 2018] and medicine [Denisova et al. 2023; Hofmann et al. 2020].

Unlike previous works [Firmino et al. 2022; Gu et al. 2022; Yang 2024], our method does not rely on deep learning techniques and therefore does not require re-training when the denoising algorithm changes. In contrast to current algorithms, which rely on blending images from the same iteration, ours allows for the blending of images acquired at varying sample counts. This eliminates the trade-off between computational efficiency and visual quality, leading to improvements in both.

We compute blending weights for each pixel channel using both the MC estimate and denoised images. These weights correspond to effective or 'predicted' iteration numbers – essentially, the samples per pixel during MC simulations. This process mimics adaptive sampling *a posteriori*.

Similar to previous studies [Firmino et al. 2022; Li et al. 2012], we utilize the Stein Unbiased Risk Estimate (SURE) for estimating the mean squared error (MSE) of denoised images against the reference. It necessitates the input radiance values to adhere to a normal distribution, crucial for SURE's accuracy.

Fig. 2. Plot of the mean variance of images generated with Mitsuba (right) in logarithmic scale.

We illustrate our approach using the Mitsuba framework [Nimier-David et al. 2019] for MCPT rendering, employing two pre-trained denoising models: Intel® Open Image Denoise (OIDN) [Áfra 2023] with albedo and normal auxiliary features, and RT, a denoiser based on OIDN but operated on albedo, normal, and input mean variances [Firmino et al. 2022]. Our results are comparable with state-of-the-art deep-learning progressive denoising methods and even surpass them in some cases, as illustrated by Figure 1.

We demonstrate the retrospective feasibility of our method and advocate for its application in real-time scenarios, highlighting its potential to improve both rendering efficiency and quality. Additionally, our approach can be effectively integrated with adaptive sampling techniques, wherein weights assigned to each pixel guide the sampling process.

2 Methods

Our objective is to find the weights for unbiased (noisy) and biased (denoised) pixels at every MC iteration, such that the blending result is as close as possible to the reference pixel value. Therefore, we will solve the following optimization problem:

$$\arg\min_{W_i \geq 0} \left\| r_i(N) - \frac{I n_i(I) + W_i d_i(I)}{I + W_i} \right\| \tag{1}$$

Here, $r_i(N)$ denotes the i-th pixel of the reference obtained at iteration N; $n_i(I)$ is the i-th pixel of the noisy MC estimate obtained at iteration I, which also represents the weight of n_i; W_i denotes the unknown weight of the i-th pixel $d_i(I)$ of the denoised image, obtained by applying a denoising algorithm d to $n_i(I)$, i.e., $d_i(I) = d(n_i(I))$. We emphasize that $d_i(I)$ is the result of denoising the MC estimate $n_i(I)$, and not the result of blending obtained at iteration $(I-1)$. For brevity, we use the term "pixel," but intend "pixel radiance". Furthermore, the reasoning presented throughout the paper applies to every channel of the pixel: red, green, and blue.

On the one hand, the solution of Eq. 1 can be expressed as follows:

$$r_i(N) = \frac{I n_i(I) + W_i d_i(I)}{I + W_i} \tag{2}$$

On the other hand, for the MC simulation [Kroese and Rubinstein 2012], an unbiased MC estimate is calculated as the average of the samples $s_i(J)$, and is given by:

$$n_i(K) = \frac{\sum_{J=1}^{K} s_i(J)}{K} \implies r_i(N) = \frac{\sum_{J=1}^{N} s_i(J)}{N} \tag{3}$$

Therefore, Eq. 2 is essentially a biased MC estimate of pixel i at iteration $N = I + W_i$. This estimate approaches an unbiased MC estimate given by Eq. 3 as the biased pixel $d_i(I)$ approaches the unbiased pixel $n_i(W_i)$, obtained at iteration W_i.

In the following subsections, we will first discuss the outcome of "ideal" blending, where the reference image is known, and the problem defined by Eq. 1 can be easily solved. Next, we will analyze the approach using the convergence curves for the practical solution. Finally, we will delve into how to compute these curves in real-time alongside the approach for estimating the denoised image.

2.1 Ideal blending

Analyzing images retrospectively with access to the reference image allows for the calculation of ideal blending between biased and unbiased images.

Let $RMSE_{n_i}$ and $RMSE_{d_i}$ denote the root mean square error (RMSE) values of the MC estimate and denoised pixels, respectively, compared to the reference image. Resolving Eq. 2 for W_i, we get:

$$W_i = I \cdot \frac{r_i(N) - n_i(I)}{d_i(I) - r_i(N)} = \pm I \cdot \frac{RMSE_{n_i}}{RMSE_{d_i}} \tag{4}$$

Upon first inspection, one may decide to always choose the plus sign in Eq. 4, given that I and RMSE values are positive, and W_i should be positive by definition. However, this solution might not be optimal, given that the RMSE considers only the distance to the reference, not the direction. Thus, while the formula in Eq. 4 doesn't necessitate values of the reference pixel directly, the ambiguity in choosing the sign remains crucial, making this theoretical solution unsuitable for practical evaluation.

2.2 Convergence curves

To find a practical solution to the problem posed by Eq. 1, we consider the following optimization problem:

$$\arg\min_{W_i \geq 0} \|d_i(I) - n_i(W_i)\| \tag{5}$$

Indeed, according to the reasoning done for Eq. 2, where the biased estimate at sample count of N approaches an unbiased MC estimate at sample count of N as long as the biased value at sample count of I approaches the unbiased value at sample count of W_i, the problems defined by Eq. 1 and Eq. 5 are equivalent. It follows from the fact that if we find W_i such that the norm of the difference between $d_i(I)$ and $n_i(W_i)$ is minimized, then this minimized norm value will directly lead to minimizing the norm value in Eq. 1. Furthermore, for W_i that solves Eq. 5, the following is true:

$$\begin{aligned} \|d_i(I) - n_i(W_i) - r_i(N) + r_i(N)\| = 0 &\implies \|(d_i(I) - r_i(N)) - (n_i(W_i) - r_i(N))\| = 0 \\ \implies \|r_i(N) - d_i(I)\| = \|r_i(N) - n_i(W_i)\| &\implies MSE_{d_i(I)} = MSE_{n_i(W_i)} \end{aligned} \tag{6}$$

Here, $MSE_{d_i(I)}$ and $MSE_{n_i(W_i)}$ are the mean square errors of denoised pixel $d_i(I)$ and MC estimate $n_i(W_i)$, respectively, compared to the reference pixel.

Our approach is based on two observations. First, the mean square error of the MC estimate compared to the reference can be estimated by its mean variance, as demostrated in [Firmino et al. 2022]. Second, the mean variance of pixel values decreases as the iteration number increases, eventually converging to zero. Consequently, each mean variance value corresponds to a specific iteration. With knowledge of the convergence curve, we can determine the iteration at which a particular mean variance value was achieved. Therefore, if we have the mean variance of the denoised pixel, we can assign it the weight it merits in Eq. 2. It's important to note that this practical solution introduces a bias, because while Eq. 6 follows from Eq. 5, the reverse is not necessarily true. However, we will demonstrate that despite this relaxation, the method does indeed perform well in practice.

In a sense, our approach can be viewed as adaptive sampling *a posteriori*. After denoising, pixels with higher variance improve and receive higher weight, as if they were estimated by more samples.

Figure 2 displays the mean variance curves in logarithmic scale for 10 different scenes generated with Mitsuba. These curves offer valuable insights into the convergence behavior of the MC rendering process across various scenes, shedding light on the relationship between iteration count and variance reduction. Understanding these curves is crucial for informing our approach to denoising and adaptive sampling.

Notably, all curves in the plot can be effectively approximated by lines of the form:

$$log(v_I) = a \cdot log(I) + b \tag{7}$$

Here, v_I represents the mean variance of the entire image obtained at the I-th iteration, and a and b are the slope and intercept of the line, respectively. Note that this I is the same as the one used in Eq. 2, as the weight of an unbiased pixel corresponds to the iteration count.

Therefore, it's possible to construct the convergence curve by calculating the slope and intercept of Eq. 7.

It is important to note that the curves in Figure 2 represent the average mean variance among all image pixels. However, these curves are much more stable than those of a single pixel, as illustrated in Figure 3 of Supplementary. The jumps at initial sample counts indicate the presence of outliers, such as when a low probability path is found. This observation suggests that some filtering of the variance might be necessary before applying the reasoning done for average mean variances to the mean variance per pixel.

2.3 Easy blending

Before passing to the curves calculation, let's consider a straightforward approach to blending biased and unbiased values involving the following formula:

$$r_i = \begin{cases} n_i, & \text{if } RMSE_{d_i} > RMSE_{n_i} \\ d_i, & \text{otherwise.} \end{cases} \tag{8}$$

The method, expressed in Eq. 8, does not directly rely on the reference pixel's value. While lacking the precision of Eq. 4, which requires knowledge of exact RMSE values, easy blending offers simplicity in implementation.

Figure 3 presents the MSE of the MC estimate, its denoised counterpart, and the easy blending solution under the assumption of knowing exact values of $RMSE_{d_i}$ and $RMSE_{n_i}$. Based on the findings from Figure 3, both quantitative and visual comparisons for the easy blending solution show promise. Consequently, we will conduct a practical evaluation later on, comparing it with our proposed approach.

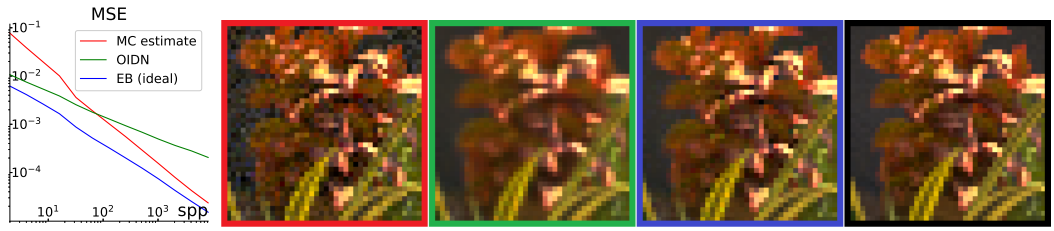

Fig. 3. Left: Plot of the mean squared error (MSE) for the Monte Carlo (MC) estimate (red), Open Image Denoise (OIDN, green), and easy blend (EB, blue) according to Eq. 8, depicting a San Miguel outdoor scene. Right: Crop of the San Miguel scene (refer to Fig. 2, middle at the bottom) showing the comparison between 128 samples per pixel (spp) MC estimate (red), OIDN (green), and EB (blue), with the 32K spp reference (black).

Note that both methods discussed above do not require the MC estimate and the denoised image to be obtained at the same iteration.

2.4 Retrospective curves calculation

Let's consider a scenario where we have the mean variance values for all rendered images from the initial iterations up to the reference image. In such cases, we can use least squares regression [Stigler 1981] to determine the parameters of Eq. 7. However, due to the lower accuracy of mean variance values in the earlier iterations, some filtering of the data is necessary (refer to Figure 3 of Supplementary).

Eq. 7 suggests a linear relationship between the points, with the mean variance decreasing as the number of samples increases. Therefore, to achieve the most accurate regression results, we prioritize mean variance values that increase from the last one. This prioritization is based on the statistical principle that the latest value is generally more accurate than previous ones. Additionally, we exclude zero-values from consideration to accommodate logarithm.

Following this filtering process, a minimum of two values is sufficient to solve Eq. 7 for the parameters a and b. It's anticipated that all pixels, except possibly completely black ones, will have an associated curve calculated retrospectively.

In the case of completely black pixels, if denoising algorithms maintain zero-values, an arbitrary weight can be assigned to denoised pixels. However, if zero-values are not maintained, denoised pixels should be assigned a zero weight.

From now on, we consider the practical solution to the blending problem, such as deciding the blending weights on-the-fly without access to the reference.

2.5 Curves calculation on-the-fly

In practice, we only have variance values for previous iterations rather than for all iterations until the reference. Despite this limitation, we can still calculate curves using the available data, although with reduced accuracy. Generally, curves calculated from earlier iterations are less accurate, with accuracy improving as the iterations progress.

Operating with this reduced data, it's possible that some pixels may not have enough information to calculate the curve at the current iteration. In such cases, we accept only the last variance value while filtering out all others. Consequently, we need to calculate the second point to enable curve calculation.

Let's consider the MC estimate formula:

$$\frac{\sum_{i=1}^{N} S_i}{N} = R \tag{9}$$

Here, N represents the number of iterations (samples) S_i for the reference R estimate. We can show that:

$$\|R - A_K\| = \left(\frac{N}{K} - 1\right) \|R - A_{N-K}\|, \tag{10}$$

where

$$A_K = \frac{\sum_{i=1}^{K} S_i}{K}, \quad A_{N-K} = \frac{\sum_{i=K+1}^{N} S_i}{N-K}, \quad 1 \le K \le N-1$$

Given that S_i are independent, we have:

$$\mathbb{E}\left[V_K\right] = \left(\frac{N}{K} - 1\right)^2 \mathbb{E}\left[V_{N-K}\right] \tag{11}$$

Here, V_K and V_{N-K} represent the mean variances at iteration K and $N-K$, respectively.

Therefore, if we have the mean variance V_K at iteration K, and we set N to be sufficiently large (e.g., 10^6), we can obtain V_{N-K} and use it for curve calculation without knowing the reference value.

However, some non-black pixels may still have zero variance in earlier iterations, and the values are generally not accurate (please refer to Figure 4 of Supplementary). To address these issues, the values should be pre-processed.

To mitigate the noise in mean variance, we apply Gaussian blur [Gedraite and Hadad 2011]. Following [Firmino et al. 2022], we set $\sigma = \min(\sqrt{\overline{V}}, 1) \times 10^2$, where $\overline{V}$ is the mean variance of the entire image. It is notable that the mean variance image exhibits high noise similar to the MC estimate. Therefore, as an alternative to Gaussian blur, we attempt to apply OIDN.

We compared the MSE between the slope and intercept of the curves calculated on-the-fly and those calculated retrospectively. These MSE values provide insights into the effectiveness of the denoising methods in preserving the accuracy of the calculated curves. Interestingly, using just two last points for curve calculation does not significantly alter the result. Moreover, the curves appear stable starting from 64 spp (see plots of Figure 4 of Supplementary).

2.6 Denoising estimation

We adopted the Stein Unbiased Risk Estimator (SURE), as proposed by Li et al. (2012) [Li et al. 2012], to assess the quality of the denoised image. [Firmino et al. 2023] demonstrated this method to be more efficient compared to the double-buffer error estimate, which was introduced by [Rousselle et al. 2012] in the context of adaptive sampling. For a single denoised pixel i, SURE can be expressed as:

$$SURE(d_i(I))) = -\sigma^2 + \|d_i(I) - n_i(I)\|^2 + 2\sigma^2 d'(n_i(I)) \tag{12}$$

where $n_i(I)$ is distributed normally with variance σ^2, and the denoised estimator d is weakly differentiable.

While most components of this equation are known, estimating the last term $d'(n_i(I))$ poses a challenge. Following [Ramani et al. 2008], we utilize the Monte Carlo SURE first-order approximation:

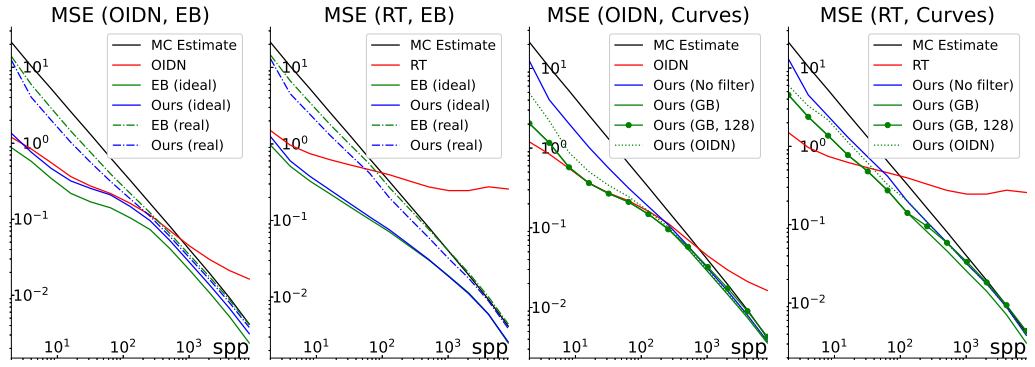

Fig. 4. Two left plots depict the retrospective mean MSE for ten test scenes using easy blending (EB) and blending based on Eq. 2 (Ours), contrasted with the MC estimate and the denoised counterpart (OIDN and RT). The evaluation is conducted using both actual errors (ideal) and error estimates (real). Two right plots present MSE results obtained by calculating the convergence curves on-the-fly using the last two points without filtering error estimates [Ours (No Filter)], applying Gaussian blur [Ours (GB)], and with OIDN [Ours (OIND)]. Additionally, the curves labeled as [Ours (GB, 128)] represent the scenario where denoising stops at the sample count of 128, with error estimates filtered using Gaussian blur.

$$d'(n_i(I)) \approx \frac{1}{\epsilon K} \sum_{k=1}^{K} b_k (d(n_i(I) + \epsilon b_k) - d_i(I))) \tag{13}$$

Here, b_k are normally distributed randoms with mean zero and covariance 1. Regarding K value, we experimentally found that setting it to 1 (for OIDN) or 4 (for RT) is sufficient. A higher count for K indicates lower stability of the denoising algorithm. As per Ramani [Ramani et al. 2008], the selection of ϵ is stated to have no impact on the outcome but is recommended to be "quite small." However, in practice, the choice of this parameter can be pivotal. Notably, the sensitivity to variance of the OIDN denoiser is twice as high on CPU compared to GPU. Through experimentation, we discovered that setting $\epsilon = 1$ yields more stable outcomes.

Due to the high variance observed at low sample counts, SURE values may display instability. Additionally, as indicated by Eq. 12, SURE values can potentially be negative, which lacks a physical interpretation since SURE is intended to estimate the mean square distance. During our experiments, we observed that setting negative SURE values to zero improves estimation accuracy at low sample counts. However, this trend shifts at higher sample counts. Additionally, setting SURE to zero at higher sample counts may result in assigning excessive weight to the denoised counterpart, potentially leading to errors if denoising is halted from a certain sample count onwards.

Indeed, it makes practical sense. We calculated the weights by first setting SURE to zero and then to its magnitude at every MC iteration. Then, we compared the blended values, according to Eq. 2, with the reference. The plot in Figure 5 of Supplementary illustrates the percentage at each sample count of how many blended values were closer to the reference by choosing zero or magnitude for negative SURE. Notably, setting negative SURE values to zero results in significantly better estimations at lower sample counts, but this tendency decreases as the sample count grows. Although it may seem at first glance that zeroing the negative SURE values produces better results, caution is needed when denoising stalls at certain iterations. In such cases, the zeroing approach can result in assigning a very high weight to biased pixels, whose quality remains constant as sample counts increase, while the quality of unbiased input improves. Therefore, it is safer to use

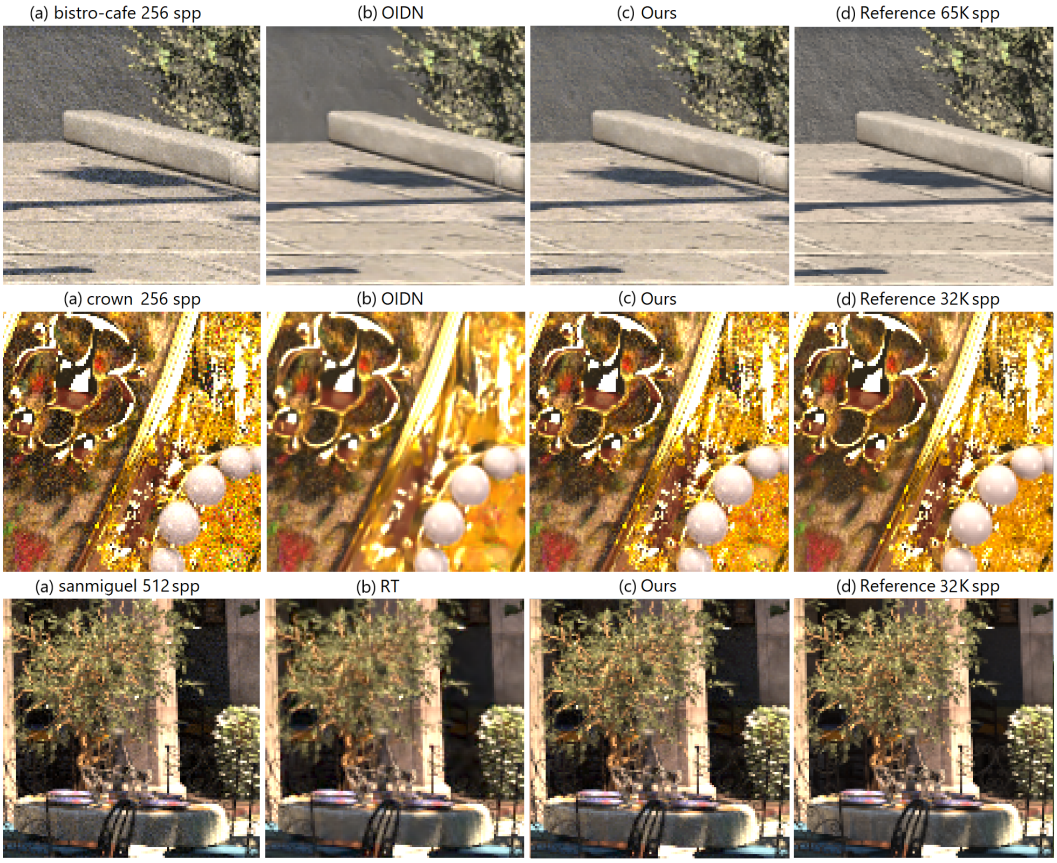

Fig. 5. Visual comparisons of the MC estimate (a) against results generated by OIDN and RT denoisers (b), and our method of curves (c) relative to the reference (d). In our method, Gaussian blur was applied for error estimates filtering.

the magnitude strategy, allowing unbiased pixels to compete with biased ones. Consequently, we switch from the zeroing strategy to the magnitude strategy when the mean variance of the MC estimate becomes less than the mean SURE.

To enhance SURE stability, we apply Gaussian blur ($w = 11$, $\sigma = min(\sqrt{\overline{V}}, 1) \times 10^2$) as we already do for the mean variance. Additionally, as an alternative approach, we also apply OIDN.

Notably, the Gaussian blur effect is constrained by the choice of σ. Meanwhile, OIDN can sometimes yield inferior results at higher sample counts compared to unfiltered values. Applying the same principle applied for Gaussian blur, we halt the denoising of mean variance and SURE when the mean variance of the entire image falls below the mean SURE.

With these components in place, we calculate the weight of the denoised pixel for blending defined by Eq. 2 using:

$$log(W_i) = \frac{log(SURE_i) - b}{a}, \tag{14}$$

where a and b define the convergence curve given by Eq. 7.

The convergence of the blending result directly follows from the method's properties. With increasing sample count, the accuracy of the mean variances of MC estimates improves, leading to

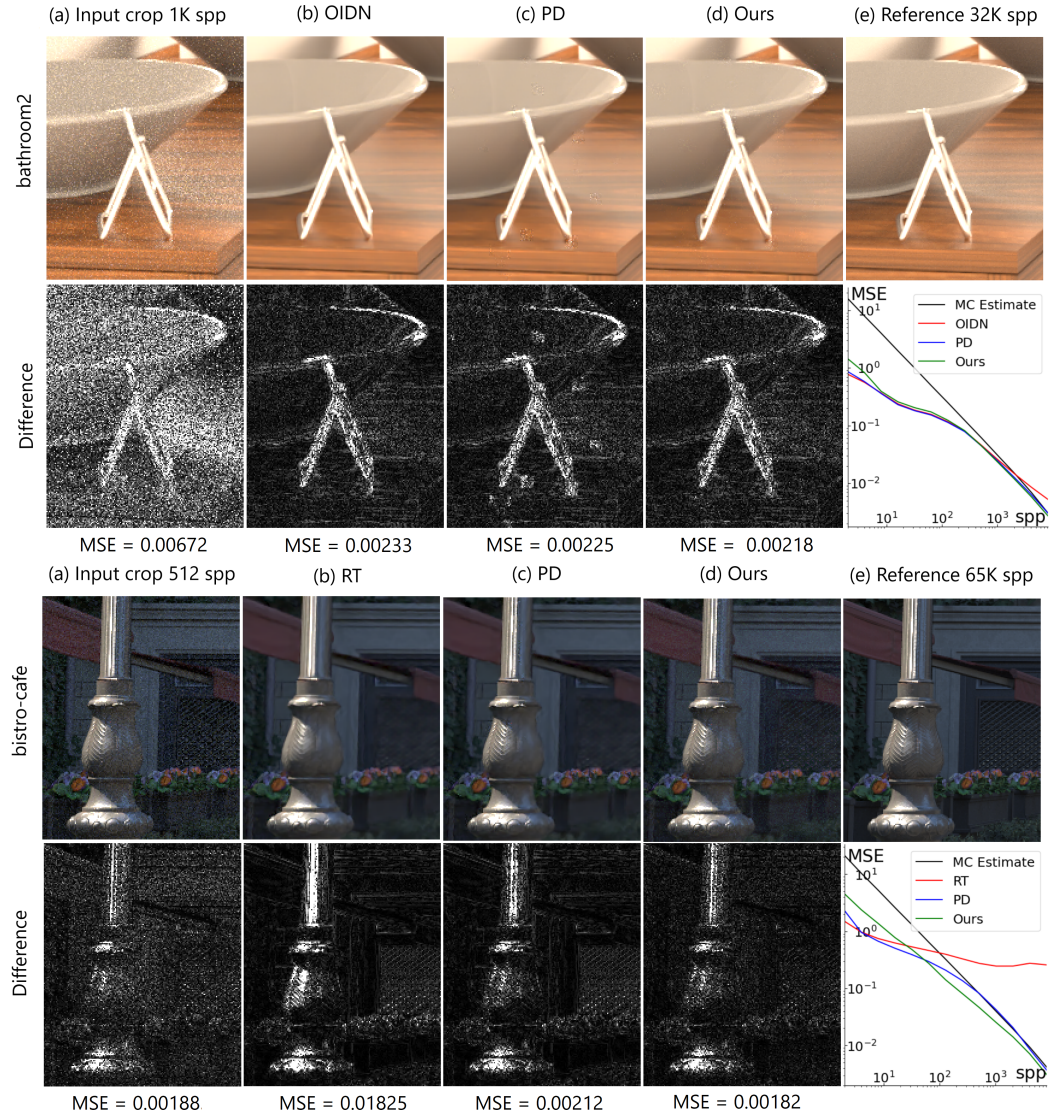

Fig. 6. Visual comparisons of the MC estimate (a) alongside results generated by our method of curves (d), denoising algorithms (b), and PD (c). Difference is a visual representation of MSE relative to the reference (e). The plots represent the MSE of entire image meanwhile MSE values under the crop represent the MSE of the crop.

more reliable estimates for blending. Furthermore, the distribution of the sample mean approaches a normal distribution [Li et al. 2012], which is a well-suited condition for applying SURE for optimal weighting. Since SURE is an unbiased estimator, it minimizes the risk of the blended image, ultimately leading to convergence of the blending process.

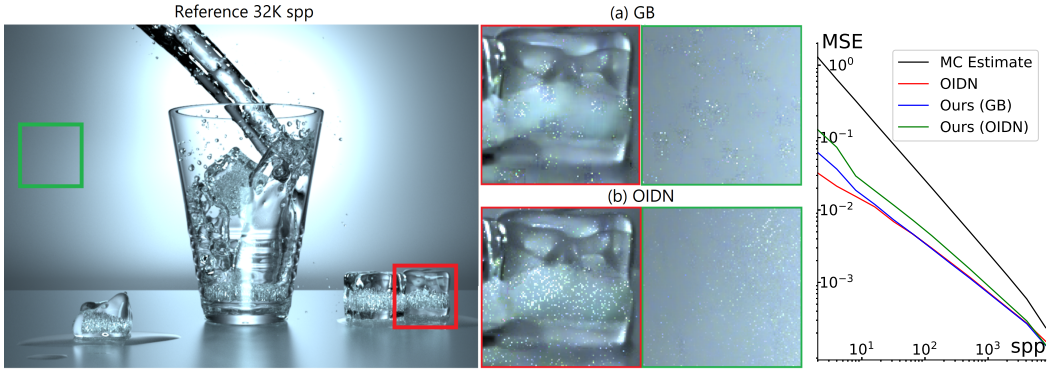

Fig. 7. Visual comparison of results produced by our method applying Gaussian blur (GB, a) and OIDN (b) to the error estimates. OIDN was employed for denoising MC estimate at a sample counts of 8. The plot illustrates the quantitative comparison in terms of MSE.

3 Results

We present our findings based on ten scenes rendered at sample counts ranging from 2 to 65536 samples per pixel, as depicted in Figure 2. These scenes are sourced from publicly available pbrt [Pharr et al. 2023] scenes and were rendered in Mitsuba [Nimier-David et al. 2019] by Firmino et al. [Firmino et al. 2022].

All considerations are based on analyzing the scenes rendered at powers of 2, from 2 to 8192 samples per pixel, together with their corresponding albedo, normal, and mean variance. We do not include the images rendered at 16384 spp as not all scenes are provided with the rendering at this sample counts. The scenes rendered at 32K and 65K spp are used as references. The references themselves are excluded from the analysis, as the metrics measurement would be meaningless for these images. This is because the reference image is being compared to itself, yielding no difference.

We evaluated our method using two denoising algorithms: OIDN with albedo and normal, and RT – a denoiser based on OIDN but operated on albedo, normal, and input mean variances [Firmino et al. 2022].

Two left plots of figure 4 illustrates the mean MSE retrospectively calculated for all test scenes using easy blending and blending according to Eq. 2, compared with the MC estimate and the denoised counterpart. The "ideal" curves correspond to blended images obtained using retrospectively calculated curves and estimating the denoised image with actual MSE with the reference images. The "real" curves represent blended images obtained using mean variances and SURE. Despite the easy blending curves being better than the curves according to Eq. 2 when calculated retrospectively, the situation changes when blended using error estimates, demonstrating that the curves approach outperforms easy blending for real scenarios.

Two right plots of Figure 4 showcases the results achieved by calculating the convergence curves on-the-fly using only the last two points, with variances both unfiltered and filtered through Gaussian blur, as well as with OIDN. Similarly, SURE values were examined without any filtering, filtered with Gaussian blur, and processed with OIDN. Additionally, the plots depict the scenario where denoising halts at the sample count of 128, with both mean variance and SURE values filtered using Gaussian blur. Evidently, the filtering substantially enhances the results, and even in cases where denoising terminates at a lower sample count, the curve smoothly aligns with the reference alongside the MC estimate.

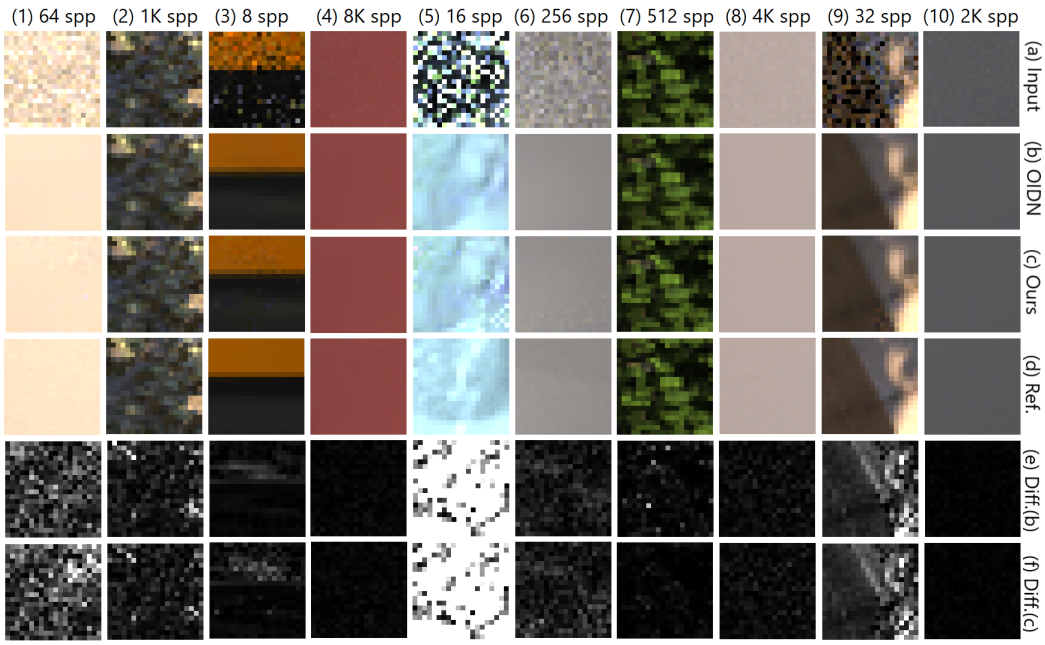

Fig. 8. Visual comparison of OIDN (b) and our method (c), evidenced by difference (e) and (f). The presented images are 20x20 crops of the exact center of Mitsuba scenes: (1) bathroom2, (2) bistro-cafe, (3) coffee, (4) crown, (5) glass-of-water, (6) kitchen, (7) landscape, (8) living-room-3, (9) sanmiguel, (10) spaceship. Inputs (a) are rendered at randomly assigned sample per pixel (spp). References (d) are rendered at 32K or 64K spp.

We performed a spectral analysis for the Crown scene, including our method with the curves calculated on-the-fly and the estimates filtered with Gaussian blur. As Figure 2 of Supplementary demonstrates, our method successfully reduces noise at the lower sample counts while maintaining the amplitudes of the higher frequencies at the higher sample counts, in contrast to the OIDN, which denoises too aggressively.

Figure 5 provides a comprehensive visual comparison of the effectiveness of our method in contrast to OIDN and RT denoisers. The visual comparison showcases how our method excels in enhancing the level of detail while effectively preserving denoising in areas where it is necessary. For additional examples, readers are encouraged to refer to Supplementary Figure 7.

Furthermore, we delve deeper into the comparative analysis of the results obtained from our method, denoising algorithms, and progressive denoising (PD) technique [Firmino et al. 2022]. Figure 6 gives a visual comparison of our method with OIDN, RT, and PD. Through a graphical representation of MSE concerning the reference image, the plots illustrate the quantitative differences between the methods. We observe comparable results when using OIDN for image denoising, while our method outperforms progressive denoising when using RT, particularly starting from 64 samples per pixel. The comparison scene by scene can be found in Supplementary Figure 8.

On our Nvidia RTX A4000, denoising an image of 1920x1080 using OIDN takes approximately 15 ms, while SURE calculation takes about 16 ms for $K = 1$ for the pre-generated randoms b_k used in Eq. 13. Curves calculation using two points and eventual Gaussian blur takes less than 1 ms. In total, producing the blending result takes about 35 ms.

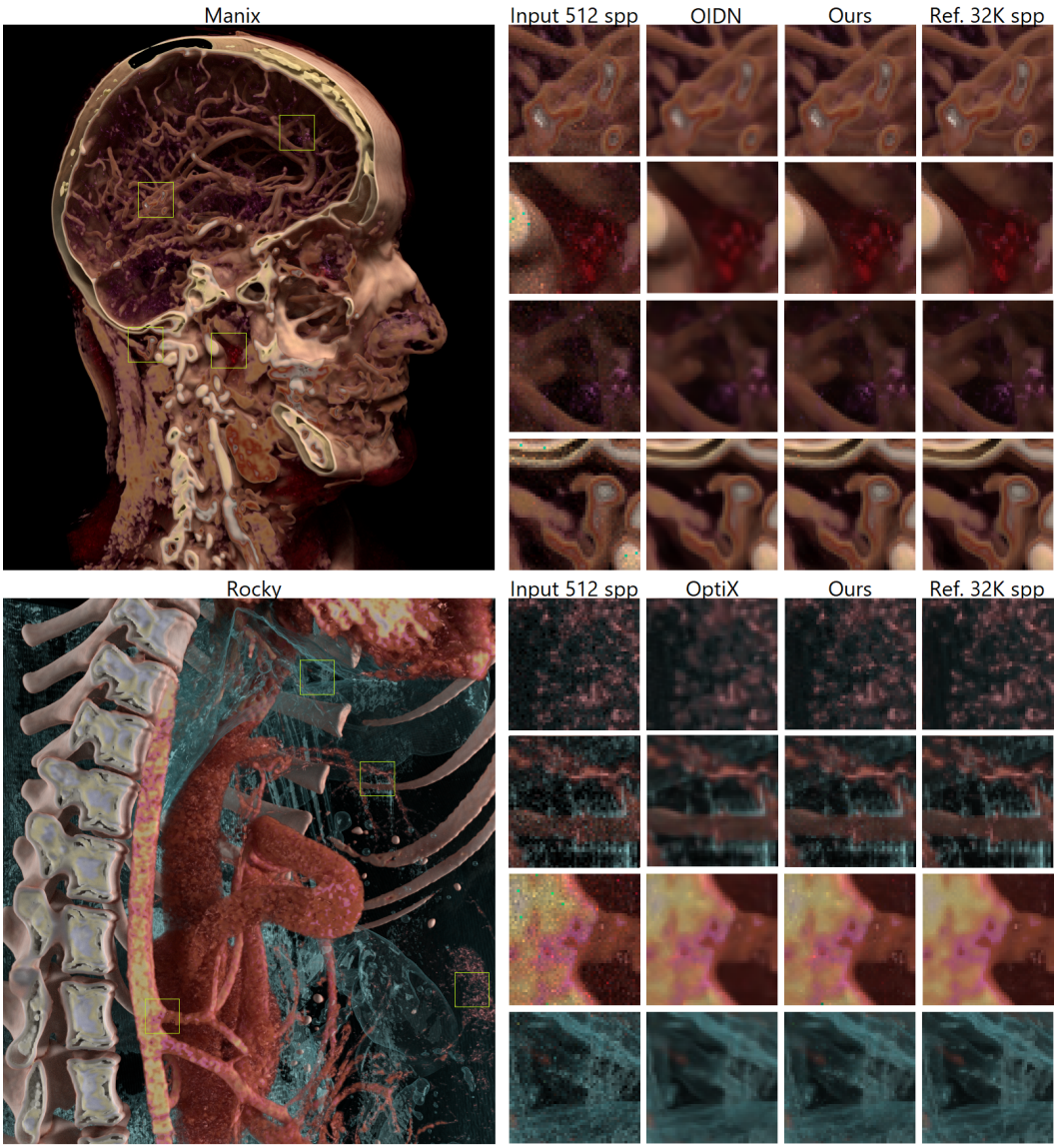

Fig. 9. Visual comparisons of the MC estimate inputs obtained at 512 spp against results generated by OIDN and OptiX, and our method relative to the reference at 32K spp. In our method, Gaussian blur was applied for error estimates filtering. Scenes were generated with AR^{2}T framework.

4 Discussion, Limitations, and Future Work

The results showcase that leveraging denoising of the mean variance and SURE yields quantitative outcomes comparable to, or even superior to, those achieved by the PD network, as evidenced in Fig. 6. However, our method requires a reasonable number of iterations of MCPT for computation. First, because the input radiance values of MC images should adhere to a normal distribution, crucial for SURE's accuracy, and the accuracy of the estimate increases with the sample count

[Li et al. 2012]. Second, because of the accuracy of mean variances needed for curves calculation. Theoretically, our method can be applied at any iteration but it starts excelling from sample counts 16 in our experiments. Applying Gaussian blur to mean variance and SURE proves advantageous for reducing estimation errors at lower sample counts. Denoising SURE with OIDN yields inferior quantitative results, although visually, the blended images appear more natural and aesthetically pleasing, exhibiting less artificiality [Jung et al. 2015], as illustrated on Fig. 7. This misleading result may be explained by the fact that the MSE metric is too basic to capture visual perception. To address this gap, we considered HDR-ꟻLIP – an error metric designed to evaluate the perceptual quality of rendered high dynamic range images [Andersson et al. 2021].

As Figure 10 of Supplementary shows, the ꟻLIP scores are generally lower for OIDN at lower sample counts. The curves representing our method with Gaussian blur on estimates overlap with that of PD (up to spp 128), becoming better at higher sample counts (from spp 256). However, our method with OIDN on estimates performs worse than the one with Gaussian blur, even in terms of average ꟻLIP scores. We believe this discrepancy may also be due to the inherent limitations of the metrics we considered: these metrics evaluate a single image and not the temporal sequence experienced during progressive rendering.

As demonstrated on Figure 8, our method is not deprived of artifacts similar to deep-learning based approach [Firmino et al. 2022]. It is most obvious at low sample counts and on homogeneous areas as exemplified by scene crops (3) and (9). Instead, in highly detailed areas like exemplified by crops (2) and (7), our approach performs particularly well. However, considering the presence of artifacts at lower sample counts and the MSE/ꟻLIP score results, it may be prudent to start applying blending techniques from a sample count determined empirically. Alternatively, blending could be applied when the mean variance of the MC estimate becomes less than the mean SURE, though the practical benefit of this approach needs to be proven.

The successful on-the-fly calculation of curves is evident, with the primary challenge lying in precise SURE calculation – better precision leading to superior results. The denoising estimation proposed in [Firmino et al. 2023] is expected to further enhance outcomes. Given OIDN's promising performance in variance and SURE denoising, a dedicated denoising algorithm for the estimators could be explored.

We demonstrated our results using OIDN and OIDN-derived denoisers. While this might seem like a limitation, our method is not dependent on the choice of the denoising algorithm, except for the requirement that the denoising operator should be weakly differentiable, as imposed by SURE. For those interested in further comparisons, we provide quantitative results of our method applied to the OptiX denoiser [Parker et al. 2010] in Figure 11 of Supplementary. Despite OptiX having different characteristics from OIDN, the results are congruent with those obtained for OIDN and RT. Applying our method to non-deep-learning-based denoisers, such as Spatiotemporal Variance-Guided Filtering [Schied et al. 2017] or its derivatives [Zhdan 2021], would undoubtedly be interesting investigation.

While our results are promising, we acknowledge that we primarily demonstrated the method on scenes generated with the Mitsuba framework. However, our approach is not limited to the specific MCPT framework. In fact, it only requires the samples to follow a normal distribution. To showcase this broader applicability, we also successfully applied our method to two scenes generated with the AR^{2}T framework [Denisova et al. 2023] from computed tomography (CT) scans: a cone beam CT of Rocky dog torso available on Kaggle https://www.kaggle.com/datasets/imaginar2t/cbctdata, and a spiral CT of a Manix head available at https://public.sethealth.app/manix.raw.gz. The trend of improved visual quality with comparable or better MSE and ꟻLIP scores observed in the Mitsuba framework is maintained when applied to the AR^{2}T scenes, as shown in Figure 9 (qualitative) and Supplementary Figure 9 (quantitative).

While our results are demonstrated on pre-generated scenes, considerations for real-time MC rendering are crucial. Firstly, determining the optimal number of mean variance buffers for curve calculation is essential. Although our experiments utilized mean variances at various sample counts, practical applications may necessitate different steps and intervals for curve updates. Secondly, exploring indicators for denoising cessation, such as the mean variance of the entire image becoming lower than the mean of SURE, could be insightful.

Additionally, our approach, applied to adaptive sampling techniques where weights assigned to each pixel can guide the process, needs practical implementation. Moreover, many sampling algorithms (not necessarily denoising algorithms) need to determine if their estimates are sufficient, not only for adaptively selecting new samples but also for combining two estimates with different amounts of variance using an appropriate weight. For example, in Reservoir-based Spatio-Temporal Importance Resampling (ReSTIR) [Bitterli et al. 2020], calculating blending weights based on variance estimation can enhance the process by providing a more robust mechanism for importance resampling. We believe that our method could find applications in these areas.

5 Conclusion

In this study, we presented a method to predict the weight of denoised images during progressive rendering. This process involves 'skipping' a specific number of MCPT iterations, determined through denoising. We dynamically adjust these weights for each pixel based on denoised variance, effectively simulating adaptive sampling a posteriori. A notable innovation of our approach is an analytical method ensuring precise convergence of the blended output to the reference image. Importantly, our method stands out for its independence from deep-learning techniques, enabling effortless integration with any denoising algorithm. We illustrate that our method can be seamlessly incorporated into pre-trained or analytical denoisers, enhancing visual quality and allowing for blending between noisy and denoised images, even those obtained at different MCPT iterations. This dual advantage not only streamlines the rendering process but also enhances efficiency and output fidelity.

The source code of our implementation is available on GitHub https://github.com/kruunua/MCPTBlender for further exploration and use.

Acknowledgments

We would like to express our gratitude to Arturo Firmino for providing valuable insights regarding error estimations and for generously making their code and Mitsuba scenes publicly available. We are also grateful to the HPG reviewers who contributed to the quality of this work with their considerations and advice. This work is partially supported by Epica International and Imaginalis s.r.l.

References

Attila T. Áfra. 2023. Intel® Open Image Denoise. https://www.openimagedenoise.org.

Pontus Andersson, Jim Nilsson, Peter Shirley, and Tomas Akenine-Möller. 2021. Visualizing Errors in Rendered High Dynamic Range Images. In *Eurographics Short Papers*. https://doi.org/10.2312/egs.20211015

Steve Bako, Thijs Vogels, Brian McWilliams, Mark Meyer, Jan Novák, Alex Harvill, Pradeep Sen, Tony Derose, and Fabrice Rousselle. 2017. Kernel-predicting convolutional networks for denoising Monte Carlo renderings. *ACM Trans. Graph.* 36, 4 (2017), 97–1.

Benedikt Bitterli, Chris Wyman, Matt Pharr, Peter Shirley, Aaron Lefohn, and Wojciech Jarosz. 2020. Spatiotemporal reservoir resampling for real-time ray tracing with dynamic direct lighting. *ACM Transactions on Graphics (TOG)* 39, 4 (2020), 148–1.

Mike Reis Bueno, Carlos Estrela, José Mauro Granjeiro, Matheus Rodrigues de Araújo Estrela, Bruno Correa Azevedo, and Anibal Diogenes. 2021. Cone-beam computed tomography cinematic rendering: clinical, teaching and research applications. *Brazilian oral research* 35 (2021).

Per Christensen, Julian Fong, Jonathan Shade, Wayne Wooten, Brenden Schubert, Andrew Kensler, Stephen Friedman, Charlie Kilpatrick, Cliff Ramshaw, Marc Bannister, et al. 2018. Renderman: An advanced path-tracing architecture for movie rendering. *ACM Transactions on Graphics (TOG)* 37, 3 (2018), 1–21.

Robert L Cook. 1986. Stochastic sampling in computer graphics. *ACM Transactions on Graphics (TOG)* 5, 1 (1986), 51–72.

Elena Denisova, Leonardo Manetti, Leonardo Bocchi, and Ernesto Iadanza. 2023. AR2T: Advanced Realistic Rendering Technique for Biomedical Volumes. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*. Springer, 347–357.

William Donnelly, Alan Wolfe, Judith Bütepage, and Jon Valdés. 2024. FAST: Filter-Adapted Spatio-Temporal Sampling for Real-Time Rendering. *Proceedings of the ACM on Computer Graphics and Interactive Techniques* 7, 1 (2024), 1–16.

Lars C Ebert, Wolf Schweitzer, Dominic Gascho, Thomas D Ruder, Patricia M Flach, Michael J Thali, and Garyfalia Ampanozi. 2017. Forensic 3D visualization of CT data using cinematic volume rendering: a preliminary study. *American Journal of Roentgenology* 208, 2 (2017), 233–240.

Arthur Firmino, Jeppe Revall Frisvad, and Henrik Wann Jensen. 2022. Progressive denoising of Monte Carlo rendered images. In *Computer Graphics Forum*, Vol. 41. Wiley Online Library, 1–11.

Arthur Firmino, Jeppe Revall Frisvad, and Henrik Wann Jensen. 2023. Denoising-Aware Adaptive Sampling for Monte Carlo Ray Tracing. In *ACM SIGGRAPH 2023 Conference Proceedings*. 1–11.

Estevão S Gedraite and Murielle Hadad. 2011. Investigation on the effect of a Gaussian Blur in image filtering and segmentation. In *Proceedings ELMAR-2011*. IEEE, 393–396.

Jeongmin Gu, Jose A Iglesias-Guitian, and Bochang Moon. 2022. Neural James-Stein combiner for unbiased and biased renderings. *ACM Transactions on Graphics (TOG)* 41, 6 (2022), 1–14.

Nikolai Hofmann, Jana Martschinke, Klaus Engel, and Marc Stamminger. 2020. Neural denoising for path tracing of medical volumetric data. *Proceedings of the ACM on Computer Graphics and Interactive Techniques* 3, 2 (2020), 1–18.

Cheolkon Jung, Tian Sun, and Aiguo Gu. 2015. Content adaptive video denoising based on human visual perception. *Journal of Visual Communication and Image Representation* 31 (2015), 14–25.

James T. Kajiya. 1986. The Rendering Equation. In *Proceedings of the 13th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH '86)*. Association for Computing Machinery, New York, NY, USA, 143–150. https://doi.org/10.1145/15922.15902

Dirk P Kroese and Reuven Y Rubinstein. 2012. Monte carlo methods. *Wiley Interdisciplinary Reviews: Computational Statistics* 4, 1 (2012), 48–58.

Tzu-Mao Li, Yu-Ting Wu, and Yung-Yu Chuang. 2012. SURE-based optimization for adaptive sampling and reconstruction. *ACM Transactions on Graphics (TOG)* 31, 6 (2012), 1–9.

Merlin Nimier-David, Delio Vicini, Tizian Zeltner, and Wenzel Jakob. 2019. Mitsuba 2: A retargetable forward and inverse renderer. *ACM Transactions on Graphics (TOG)* 38, 6 (2019), 1–17.

Steven G Parker, James Bigler, Andreas Dietrich, Heiko Friedrich, Jared Hoberock, David Luebke, David McAllister, Morgan McGuire, Keith Morley, Austin Robison, et al. 2010. Optix: a general purpose ray tracing engine. *Acm transactions on graphics (tog)* 29, 4 (2010), 1–13.

Matt Pharr, Wenzel Jakob, and Greg Humphreys. 2023. *Physically based rendering: From theory to implementation*. MIT Press.

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 (2008), 1540–1554.

Fabrice Rousselle, Claude Knaus, and Matthias Zwicker. 2012. Adaptive rendering with non-local means filtering. *ACM Transactions on Graphics (TOG)* 31, 6 (2012), 1–11.

Christoph Schied, Anton Kaplanyan, Chris Wyman, Anjul Patney, Chakravarty R Alla Chaitanya, John Burgess, Shiqiu Liu, Carsten Dachsbacher, Aaron Lefohn, and Marco Salvi. 2017. Spatiotemporal variance-guided filtering: real-time reconstruction for path-traced global illumination. In *Proceedings of High Performance Graphics*. 1–12.

Peter Shirley and R Keith Morley. 2008. *Realistic ray tracing*. AK Peters, Ltd.

Stephen M. Stigler. 1981. Gauss and the Invention of Least Squares. *The Annals of Statistics* 9, 3 (1981), 465 – 474. https://doi.org/10.1214/aos/1176345451

Yuqi Yang. 2024. A denoising model for MC rendered images based on fusion kernel prediction and generation of adversarial networks. In *Fourth International Conference on Computer Vision and Data Mining (ICCVDM 2023)*, Vol. 13063. SPIE, 391–399.

Dmitry Zhdan. 2021. ReBLUR: A Hierarchical Recurrent Denoiser. *Ray Tracing Gems II: Next Generation Real-Time Rendering with DXR, Vulkan, and OptiX* (2021), 823–844.