Recursive Control Variates for Inverse Rendering

BAPTISTE NICOLET, École Polytechnique Fédérale de Lausanne (EPFL) and NVIDIA, Switzerland
FABRICE ROUSSELLE, NVIDIA, Switzerland
JAN NOVÁK, NVIDIA, Czech Republic
ALEXANDER KELLER, NVIDIA, Germany
WENZEL JAKOB, École Polytechnique Fédérale de Lausanne (EPFL), Switzerland
THOMAS MÜLLER, NVIDIA, Switzerland

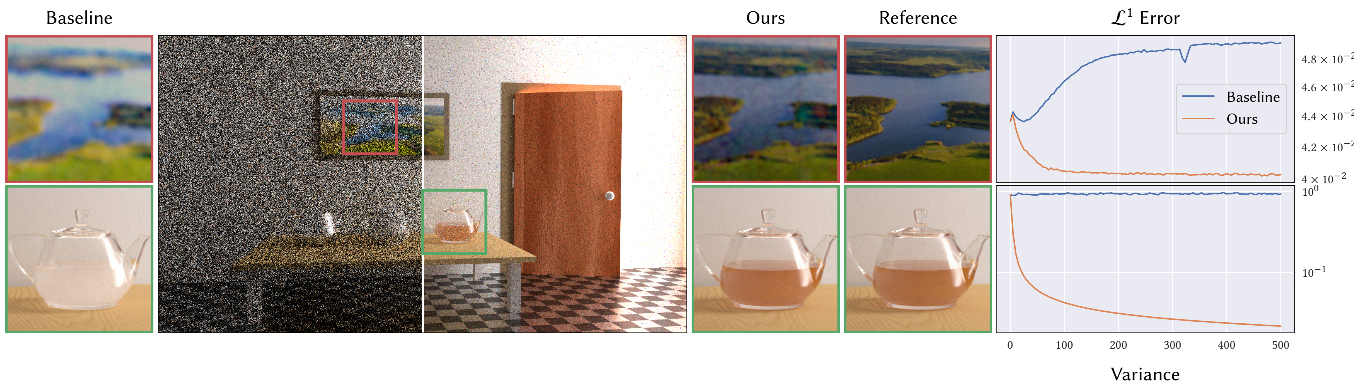


Fig. 1. We improve physically based differentiable rendering by introducing a recursive control variate into the optimization loop. Left: a baseline method [Vicini et al. 2021] repeatedly renders noisy images to compute gradients. The noise leads to poor convergence. Right: applied on top of the baseline, our recursive control variate reduces the noise using information from similar renderings in prior optimization steps. This leads to a reconstruction that is much closer to the reference image. The plots show the evolution of the loss function and rendering variance as the optimization progresses.

We present a method for reducing errors—variance and bias—in physically based differentiable rendering (PBDR). Typical applications of PBDR repeatedly render a scene as part of an optimization loop involving gradient descent. The actual change introduced by each gradient descent step is often relatively small, causing a significant degree of redundancy in this computation. We exploit this redundancy by formulating a gradient estimator that employs a *recursive control variate*, which leverages information from previous optimization steps. The control variate reduces variance in gradients, and, perhaps more importantly, alleviates issues that arise from differentiating loss functions with respect to noisy inputs, a common cause of drift to bad local minima or divergent optimizations. We experimentally evaluate our approach on a variety of path-traced scenes containing surfaces and volumes and observe that primal rendering efficiency improves by a factor of up to 10.

CCS Concepts: • **Computing methodologies** → **Rendering**.

Authors' addresses: Baptiste Nicolet, École Polytechnique Fédérale de Lausanne (EPFL) and NVIDIA, Switzerland, baptiste.nicolet@epfl.ch; Fabrice Rousselle, NVIDIA, Switzerland, frousselle@nvidia.com; Jan Novák, NVIDIA, Czech Republic, jnovak@nvidia.com; Alexander Keller, NVIDIA, Germany, akeller@nvidia.com; Wenzel Jakob, École Polytechnique Fédérale de Lausanne (EPFL), Switzerland, wenzel.jakob@epfl.ch; Thomas Müller, NVIDIA, Switzerland, tmueller@nvidia.com.

Additional Key Words and Phrases: Differentiable rendering, control variates

ACM Reference Format:
Baptiste Nicolet, Fabrice Rousselle, Jan Novák, Alexander Keller, Wenzel Jakob, and Thomas Müller. 2023. Recursive Control Variates for Inverse Rendering. *ACM Trans. Graph.* 42, 4, Article 1 (August 2023), 13 pages. https://doi.org/10.1145/3592139

1 INTRODUCTION

Inverse rendering—the reconstruction of a virtual 3D scene from images—has become a mainstream tool for content capture and creation. Differentiable renderers, in which the rendering function is inverted by gradient descent, rely on a costly optimization loop: for each gradient descent step, of which there may be thousands, the algorithm has to render an image of the current state of the reconstruction ("primal" rendering) and compute gradients with respect to scene parameters ("adjoint" rendering). Yet, consecutive gradient descent steps often correspond to minuscule changes in the scene, meaning that a large portion of the rendering computation is redundant.

For this reason, it is common to use only a few samples per pixel, resulting in noisy primal rendering and noisy gradients. After all, optimizers like stochastic gradient descent are designed to deal with noisy gradients, and small noisy steps permit a more fine-grained exploration of the parameter space.

However, a subtle aspect that is often neglected is that the optimizer is not directly applied to the noisy image, but to a loss computed from it. Many common loss functions introduce bias when differentiated with respect to noisy inputs, even when the images are themselves unbiased. Affected are all non-$\mathcal{L}^2$ loss functions, including the frequently used $\mathcal{L}^1$ error. Optimizations using learned perceptual losses such as the widely used VGG loss [Johnson et al. 2016] tend to break down entirely when provided with poorly converged images. Prior work has used tens to hundreds of samples per pixel [Nimier-David et al. 2022, 2019] to alleviate bias-related issues at a considerable computational cost.

Our work leverages the redundancy in the primal rendering phase by using prior renderings as control variates. Control variates [Fieller and Hartley 1954; Kalos and Whitlock 1986] are a generic, unbiased variance reduction technique for Monte Carlo integration. They are particularly appealing in the context of inverse rendering since gradient descent naturally leads to high correlation between subsequent optimization steps, and the noise reduction is proportional to this high correlation.

To this end, we propose a *recursive* control variate that combines *all* prior renderings, weighted such that its correlation with the next rendering, and thus the noise reduction, is maximized. To determine the aforementioned weighting, we derive a recurrence relation in terms of the statistics of our estimators, which we in turn estimate with a exponentially weighted variant of Welford's algorithm [1962].

It is imperative that the derivative estimate has low bias, as it can cause gradient descent to drift to undesirable local minima or even diverge. The recursive scheme overcomes the need for the large number of samples in rendering. In fact, when the optimization reaches steady state, our recursive control variate becomes *equivalent* to rendering with a correspondingly larger sample count. Compared to prior work, we report up to 10× higher rendering efficiency, leading to either a significant speedup at equal reconstruction quality or higher reconstruction quality after an equal amount of time; see Figure 1.

Our paper starts by reviewing related work (Section 2) followed by the derivation of our recursive control variate (Section 3). Section 4 presents an investigation of the sources of bias in differentiable rendering. We conclude with a thorough evaluation (Section 6) and discussion (Section 7) of our method.

2 RELATED WORK

Our method is related to differentiable rendering, temporal reuse in graphics, and the theory of control variates. The following paragraphs review relevant prior work in all three areas.

Differentiable rendering. The term differentiable rendering can refer to a variety of different techniques including rasterization [Hasselgren et al. 2021; Kato et al. 2017; Laine et al. 2020; Liu et al. 2019; Loper and Black 2014; Munkberg et al. 2021; Petersen et al. 2019; Rhodin et al. 2015], sphere tracing of signed distance functions [Bangaru et al. 2022; Vicini et al. 2022], ray marching through emissive volumes [Barron et al. 2021; Mildenhall et al. 2020; Müller et al. 2022; Yu et al. 2021], path tracing of globally illuminated meshes [Azinovic et al. 2019; Bangaru et al. 2020; Hasselgren et al. 2022; Li et al. 2018; Zhang et al. 2020], volume rendering [Che et al. 2020; Gkioulekas et al. 2013; Khungurn et al. 2016; Nimier-David et al. 2022; Zhang et al. 2021], and combinations thereof [Jakob et al. 2022a; Nimier-David et al. 2020, 2019; Vicini et al. 2021; Zeltner et al. 2021].

Our recursive control variates are largely agnostic to the underlying algorithm. They require only minimal changes to reduce noise, e.g., by controlling the pseudo-random number generator seed or fixing path vertices across optimization steps. Thus, our method is orthogonal and beneficial to most of the above baselines—even more so, the noisier they are. In Section 6, we demonstrate higher-quality material optimization with *Path Replay Backpropagation* (PRB) [Vicini et al. 2021] and twice faster volume optimization with *Differential Ratio Tracking* (DRT) [Nimier-David et al. 2022].

Temporal computation reuse. Practitioners have long sought to amortize the rendering cost across consecutive frames both in interactive applications and in offline rendering of animated sequences. Existing work ranges from biased screen-space approaches like temporal anti-aliasing [Karis 2014; Marrs et al. 2018] and denoising [Chaitanya et al. 2017; Hasselgren et al. 2020; Schied et al. 2018; Vogels et al. 2018], over scene-space data structures like photon mapping [Elek et al. 2012] and illumination caches [Binder et al. 2022; Majercik et al. 2022; Müller et al. 2021; Seyb et al. 2020] to unbiased techniques such as path guiding [Dittebrandt et al. 2020], reservoir resampling [Bitterli et al. 2020; Lin et al. 2022; Talbot et al. 2005], and control variates [Keller 2001; Manzi et al. 2016; Rousselle et al. 2016].

Two properties of control variates are particularly appealing in the context of differentiable rendering: they are simple to implement and provide unbounded noise reduction when the optimization reaches a steady state, which equivalent to rendering with an increasingly larger sample count. Ease of implementation implies that they can sometimes even be applied *on top of* prior work. We demonstrate a proof-of-concept combination with the OptiX denoiser [Chaitanya et al. 2017] in Figure 9.

Pertaining to differentiable rendering, concurrent work in cloud tomography [Czerninski and Schechner 2023] resamples paths across consecutive optimization steps in order to avoid the computational cost of tracing new ones. However, the authors report that this comes at the cost of *increased* noise—a reversal of the characteristics of our method.

Control variates. Training pipelines in the area of machine learning tend to rely on variants of *stochastic gradient descent* (SGD) to replace the costly gradient evaluation with a cheaper statistical estimate computed from a randomly selected subset of the training data. The resulting variance can impede convergence, which has motivated variance-reduced methods [Gower et al. 2020] including *Stochastic Average Gradient "Amélioré"* (SAGA) [Defazio et al. 2014] and *Stochastic Variance-Reduced Gradient* (SVRG) [Johnson and Zhang 2013]. Both SAGA and SVRG maintain a running approximation of the full gradient and use it to provide unbiased variance reduction using the framework of control variates. Such approaches are not compatible with differentiable rendering, where variance arises in a different manner owing to the numerical integration over continuous domains.

To reduce rendering noise, the first use of control variates is due to Lafortune and Willems [1995a,b] who fit a constant ambient lighting term and a 5D piecewise-constant tree to the spatio-directional radiance field. Because these fitted functions are analytically and cheaply integrable, they are trivially usable as control variates.

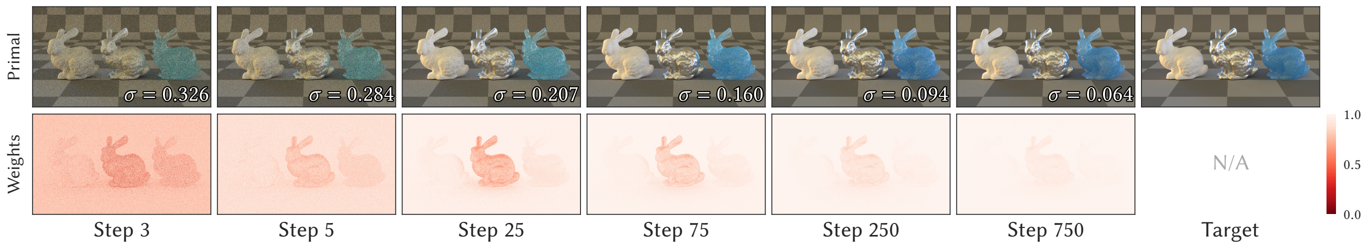


Fig. 2. Our method incrementally adapts a control weight to achieve an efficient reuse of samples from prior iterations. In this reconstruction of the albedo, roughness, and extinction coefficient of the three respective bunnies, renderings become increasingly correlated as the exploration of the parameter space stabilizes. The proposed control weight reacts to this correlation and steadily approaches unity while the primal rendering variance decreases.

Other work introduces more sophisticated control variates with known integral, such as linear combinations of probability densities [Fan et al. 2006; Kondapaneni et al. 2019], piecewise polynomial functions parameterized by a tree [Crespo et al. 2021], neural normalizing flows [Müller et al. 2020], or the diffusion profile in sub-surface scattering [Xie and Olano 2021].

Certain classical algorithms can also be viewed through the lens of control variates, such as residual tracking [Novák et al. 2014], delta tracking [Georgiev et al. 2019], multiple importance sampling [Kondapaneni et al. 2019], Monte Carlo integration [Salaün et al. 2022], and even temporal gradient-domain rendering [Manzi et al. 2016], which is equivalent to a particular (unweighted) configuration of the image space control variates of Rousselle et al. [2016].

Particularly relevant to our work are control variates that rely on successive estimation rather than analytic integration [Keller 2001; Manzi et al. 2016; Misso et al. 2022; Rousselle et al. 2016; Szécsi et al. 2004]. In our case, the light transport simulated in prior optimization steps feeds into a successively less noisy control variate for future optimization steps.

3 VARIANCE REDUCTION BY CORRELATED SAMPLING

We introduce a new generalization of control variates [Fieller and Hartley 1954] that facilitates its adoption in optimization algorithms as an error reduction technique.

3.1 Preliminaries

Control variates. We use the notation $\langle F\rangle$ to denote a Monte Carlo estimator of an integral F involving the function f. This estimator may exhibit a large amount of variance. Given another function g resembling f, whose integral G is furthermore available in closed form, the variance of estimating F can be reduced by the addition of a *control variate* [Fieller and Hartley 1954]

$$\langle F\rangle_{\mathrm{CV}} = \langle F\rangle + \alpha\big(G - \langle G\rangle\big)\,. \tag{1}$$

For this technique to be effective, the estimator $\langle G\rangle$ should be correlated with $\langle F\rangle$, which is typically accomplished by evaluating both estimators using the same set of pseudo-random numbers.

Since $\mathbb{E}[G - \langle G\rangle] = 0$, the real-valued *control weight* α may be chosen arbitrarily without introducing bias. The optimal choice that minimizes variance [Owen 2013] is given by

$$\alpha_{\mathrm{opt}} = \frac{\mathrm{Cov}[\langle F\rangle, \langle G\rangle]}{\mathrm{Var}[\langle G\rangle]}\,. \tag{2}$$

When G is not known in closed form, a Monte Carlo estimate can be used instead, which we denote by $\langle G\rangle_0$. The variance of such an estimator is minimized by the optimal control weight

$$\alpha_{\mathrm{opt}} = \frac{\mathrm{Cov}[\langle F\rangle, \langle G\rangle - \langle G\rangle_0]}{\mathrm{Var}[\langle G\rangle - \langle G\rangle_0]}\,. \tag{3}$$

Building on Equation (3), Rousselle et al. [2016] show how to reduce the variance when *re-rendering* an existing scene following a change to its parameters. Let $\langle F_0\rangle_p$ denote an initial rendering computed using a particular set of Monte Carlo samples with index p, while $\langle F_1\rangle_q$ refers to the estimator following a modification of the scene. If $p = q$, the second estimator uses the same sample set, causing it to be correlated with $\langle F_0\rangle_p$. If $p \neq q$, then p and q will be deemed statistically independent sets of samples. In general, $\langle F_i\rangle_p$ will refer to a rendering of an evolving scene at iteration i using the set of samples p.

Rousselle et al. [2016] propose to use the original rendering $\langle F_0\rangle_p$ as a control variate, resulting in the estimator

$$\langle F_1\rangle_{\mathrm{CV}} = \langle F_1\rangle_1 + \alpha\big(\langle F_0\rangle_0 - \langle F_0\rangle_1\big)\,, \tag{4}$$

where the weight

$$\alpha = \frac{\mathrm{Cov}[\langle F_1\rangle_1, \langle F_0\rangle_1]}{\mathrm{Var}[\langle F_0\rangle_0 - \langle F_0\rangle_1]} \tag{5}$$

is computed using numerical approximations of the (co-) variances. The estimator $\langle F_0\rangle_0$ does not appear in the covariance (cf. Equation (3)) because it is statistically independent from $\langle F_1\rangle_1$ and therefore does not contribute.

3.2 Recursive Control Variates

Applications of differentiable rendering tend to re-render the scene thousands of times. At the same time, the magnitude of changes between steps is relatively small—often orders of magnitude below those found in primal renderings of animated or interactive content. Our extension of the previously introduced control variate exploits this redundancy across an arbitrarily long sequence of renderings. We generalize the concept in Equation (4) to a sequence, where the new estimator

$$\langle F_n\rangle_{\mathrm{CV}} = \langle F_n\rangle_n + \alpha_n\big(\langle F_{n-1}\rangle_{\mathrm{CV}} - \langle F_{n-1}\rangle_n\big) \tag{6}$$

recursively compounds variance reduction across the entire sequence.

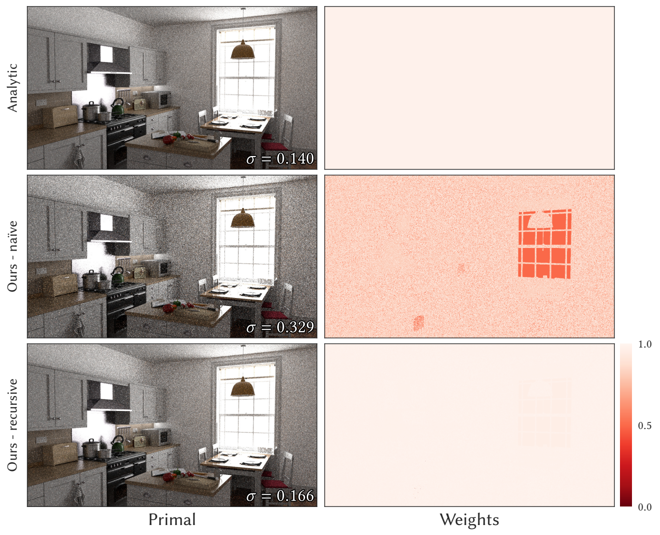


Fig. 3. Ignoring the correlation between consecutive iterates can severely impact the effectiveness of our method. In this figure, we apply our control variate to repeated rendering passes of an unchanging scene for 50 iterations, which is representative of an optimization that has reached a steady state. Optimal control weights can be computed analytically in this simple case. Top: the analytic control weight $\alpha_n = n/n{+}1$. Middle: computation of α_n via Equation (8), using Welford's algorithm to estimate the $\mathrm{Var}[\langle F_n \rangle_{\mathrm{CV}}]$ statistic. Bottom: further replacing the online-estimator with the recurrence from Equation (10). This last approach yields a better approximation of the optimal solution and achieves a comparable variance reduction.

The starting point of this sequence is set to

$$\langle F_0 \rangle_{\mathrm{CV}} := \langle F_0 \rangle_0 \tag{7}$$

and its optimal control weight is given by

$$\alpha_n = \frac{\mathrm{Cov}[\langle F_n \rangle_n, \langle F_{n-1} \rangle_n]}{\mathrm{Var}[\langle F_{n-1} \rangle_n] + \mathrm{Var}[\langle F_{n-1} \rangle_{\mathrm{CV}}]}\,. \tag{8}$$

While the structure of the denominator above does not match Equation (5), they are mathematically equivalent because $\langle F_{n-1} \rangle_n$ and $\langle F_{n-1} \rangle_{\mathrm{CV}}$ are computed from independent sample sets and because the identity $\mathrm{Var}[X - Y] = \mathrm{Var}[X] + \mathrm{Var}[Y]$ holds for independent variables X and Y. We estimate the variances separately, which motivates this definition.

The control weight α_n has an intuitive interpretation: it determines the degree to which past information can be reused. Large scene parameter changes will generally reduce the correlation between current and previous renderings, causing α_n to be nearly zero. This effectively discards the information gathered in previous iterations. Conversely, an unchanging scene is maximally correlated, in which case the numerator equals $\mathrm{Var}[\langle F_{n-1} \rangle_n]$. If *all* renderings are maximally correlated, we have $\alpha_n = n/n{+}1$. Inserting this expression into Equation (6) reveals that $\langle F_i \rangle_{\mathrm{CV}}$ turns into the mean of all preceding estimates. Figure 2 visualizes the per-pixel control weights during an inverse rendering optimization. In early iterations, the control weights start out close to zero, progressively approaching unity as more and more information is gathered. The variance of the rendered primal images decreases correspondingly.

Optimality. While the approach we just presented uses all previous renderings to reduce variance at a given iteration, it does not combine them in the theoretically optimal way, which would be to use all previous renderings as *individual* control variates:

$$\langle F_n \rangle_{\mathrm{CV}} = \langle F_n \rangle_n + \sum_{i=0}^{n-1} \alpha_i \big(\langle F_i \rangle_i - \langle F_i \rangle_n \big) \tag{9}$$

However, such an approach is not practical: determination of control weights α_i entails rendering all previous states using the same set of Monte Carlo samples to construct and process an ever-growing covariance matrix. As we observed, earlier terms become less important as the optimization evolves, hence this prohibitive computational effort would also be wasteful. Our method can be interpreted as an approximation of this strategy, whose weights are equivalent to Equation (9) when the optimization is in its steady state, as illustrated in Figure 3.

Estimating the control weight α_n. The control variate described so far remains incomplete in the sense that it still lacks an explanation of how the (co-) variance terms in Equation (8) should be computed. Unfortunately, these terms are intractable using analytic methods, which means that approximations are needed in practice. The better these approximations, the closer the control variate will approach theoretical optimality.

Our method estimates the needed terms $\mathrm{Cov}[\langle F_n \rangle_n, \langle F_{n-1} \rangle_n]$, $\mathrm{Var}[\langle F_{n-1} \rangle_n]$, and $\mathrm{Var}[\langle F_{n-1} \rangle_{\mathrm{CV}}]$ statistically, hence we seek approximations that are characterized by a small amount of variance and bias. Compared to other parts of the optimization, some bias in the statistics estimation is tolerable, since it merely impacts the effectiveness of the control variate.

Low variance and low bias are somewhat conflicting goals: the use of samples from many prior optimization steps will naturally produce more converged estimates. At the same time, long-term aggregation with an evolving scene introduces bias, as samples are no longer identically distributed. We use an exponentially weighted aggregation to compromise between the two goals. The estimates become unbiased once the optimization reaches a steady state.

The most commonly used (co-) variance estimator performs two passes through a dataset to determine the sample mean and final estimate. This is problematic in long-running optimizations, since every evaluation of α_n would need access to all prior renderings. We instead rely on Welford's algorithm [1962], which computes an equivalent estimate in a single pass using a fixed amount of temporary storage. Appendix A reviews the original form of Welford's algorithm along with exponential weighting [Schubert and Gertz 2018; Xie and Olano 2021] and a modification to avoid startup bias.

A final challenge arises when estimating the variance of $\langle F_n \rangle_{\mathrm{CV}}$: the recursive definition of our control variate introduces correlations across iterations, which violates independence assumptions in standard variance estimators.
We propose a recurrence that accounts for this effect:

$$\begin{aligned}\mathrm{Var}[\langle F_n \rangle_{\mathrm{CV}}] &= \mathrm{Var}[\langle F_n \rangle_n + \alpha_n \big(\langle F_{n-1} \rangle_{\mathrm{CV}} - \langle F_{n-1} \rangle_n \big)] \\ &= \mathrm{Var}[\langle F_n \rangle_n] + \alpha_n^2 \mathrm{Var}[\langle F_{n-1} \rangle_n] \\ &\quad - 2\alpha_n \mathrm{Cov}[\langle F_n \rangle_n, \langle F_{n-1} \rangle_n] + \alpha_n^2 \mathrm{Var}[\langle F_{n-1} \rangle_{\mathrm{CV}}]\end{aligned} \tag{10}$$

Although correct, there is a practical issue with this equation: exact statistics involving $\langle F_n \rangle$ and $\langle F_{n-1} \rangle$ are generally unknown.

We found that the recurrence remains effective when it is used in an approximate sense, by estimating statistics with the presented variant of Welford's algorithm. Figure 3 examines the effectiveness of this approximation in a setting where optimal weights can be computed analytically.

Although the control weight can in principle take on any value, e.g., $\alpha_n < 0$ to exploit negative correlation, such a situation would not be meaningful in our application that repeatedly renders the same scene with small variations. We always clamp α_n to the range $[0, 1]$, which is helpful especially at the beginning of the optimization when the statistics are not yet well-converged.

Bias. Unbiasedness of a control variate generally requires that its input estimators are not correlated with the control weight [Lavenberg et al. 1982; Nelson 1990]. This is not the case in our method, which is a key limitation requiring further discussion. Indeed, our initial definition of α_n correlates it with respect to both $\langle F_{n-1} \rangle_{\mathrm{CV}}$ and $\langle F_{n-1} \rangle_n$, which leads to noticeable bias shown in Figure 4 (left).

In practice, we introduce a small modification by updating the running (co-)variance estimates *after* evaluating $\langle F_n \rangle_{\mathrm{CV}}$. The statistics estimators thus lag one iteration behind, ensuring that α_n and $\langle F_{n-1} \rangle_{n-1}$ are statistically independent. Some bias clearly remains, as α_n is still correlated with $\langle F_{n-1} \rangle_{\mathrm{CV}}$ (both computed using a stream of previous renderings). In other words, our approach removes the correlation with the previous rendering, but not with all the ones before it. The right side of Figure 4 illustrates the impact of this change. As we will show in the next section, our method can significantly reduce the amount of bias caused by loss function derivatives. As such, we trade a small amount of bias in the control variate for a larger bias reduction in the loss function gradient.

4 BIAS IN DIFFERENTIABLE RENDERING

Derivative-based inverse rendering methods require some type of loss function $\mathcal{L}$ to quantify the difference between the simulation's output F_θ and reference data I_{ref}. Different representations of F_θ and I_{ref} are conceivable: in the simplest case, this could be a single image. More commonly, multiple images from different viewpoints must be combined to reduce ambiguity. Instead of an image, the renderer could also output a sparse sampling of the light field or a projection onto a set of basis functions. The observations in this section apply to all of these possibilities.

We usually seek the global minimizer θ^* of the composition of simulation and loss, i.e.,

$$\theta^* = \arg\min_\theta \mathcal{L}(F_\theta, I_{\mathrm{ref}}). \tag{11}$$

Due to the challenging nature of this non-linear objective, we must taper our expectations and settle for a local minimum. This normally involves gradient-based techniques based on the derivative of the objective. Following expansion via the chain rule, an estimator of this derivative is given by

$$\langle \partial_\theta \mathcal{L}(F_\theta, I_{\mathrm{ref}}) \rangle = \langle \partial_\theta F_\theta \rangle \cdot \partial \mathcal{L}(\langle F_\theta \rangle, I_{\mathrm{ref}}). \tag{12}$$

A subtle issue with this approach is that the estimate can be biased even when $\langle F_\theta \rangle$ is unbiased, which breaks traditional convergence

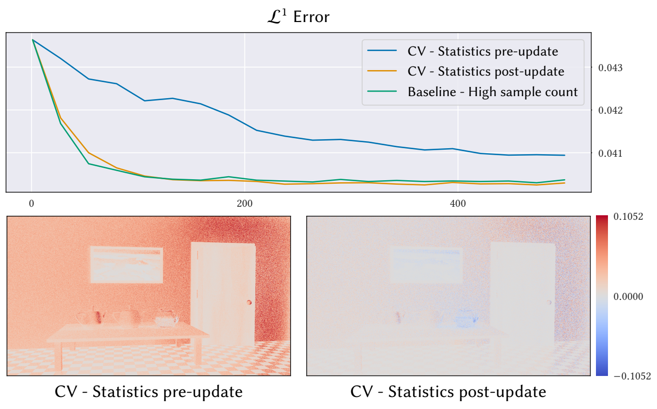


Fig. 4. Bias of the control variate estimator $\langle F_n \rangle_{\mathrm{CV}}$ visualized using the difference between the expected rendering and the CV estimate at iteration #256 of the optimization from Figure 9. Updating the statistics estimates *before* evaluating the control variate correlates the control weight with the current image, which adds significant bias (bottom left). Updating the statistics *after* evaluating the control variate removes this source of bias (bottom right). A small amount of bias remains due to the correlation between the control weight and the previous iteration's control variate. In return, variance reduction yields results with a significantly higher effective sample count (1024 vs. 16 samples per pixel) at a fraction of the cost (top).

guarantees of stochastic gradient descent (SGD). For example, SGD with averaging finds the minimum of any strongly convex objective, but it may drift or even diverge when fed with biased gradients. A first potential source of bias arises from the multiplication operation in Equation (12), which is only unbiased if both factors are independent estimators. The established way to deal with this constraint is to simply estimate $\langle F_\theta \rangle$ twice with different random variates [Gkioulekas et al. 2013]. We use path replay back-propagation (PRB) [Vicini et al. 2021] to estimate the first factor $\langle \partial_\theta F_\theta \rangle_{\mathrm{PRB}}$ and insert our recursive control variate $\langle F_\theta \rangle_{\mathrm{CV}}$ into the loss term. This results in the final form of the proposed gradient estimator:[1]

$$\boxed{\langle \partial_\theta \mathcal{L}(F_\theta, I_{\mathrm{ref}}) \rangle = \langle \partial_\theta F_\theta \rangle_{\mathrm{PRB}} \cdot \partial \mathcal{L}(\langle F_\theta \rangle_{\mathrm{CV}}, I_{\mathrm{ref}})\,.} \tag{13}$$

The second source of bias is less well studied and harder to overcome: $\partial \mathcal{L}(\langle F_\theta \rangle, I_{\mathrm{ref}})$ is only unbiased when $\langle F_\theta \rangle$ is noise-free, or when the derivative of the loss $\partial \mathcal{L}$ is a linear function, since $\mathbb{E}[f(X)] \neq f(\mathbb{E}[X])$ for general random variables X unless the function f is linear.[2] Here, f corresponds to $\partial \mathcal{L}$ and X to $\langle F_\theta \rangle$.

This observation has dire consequences: in general, the $\mathcal{L}^2$ loss is *the only loss function* that leads to unbiased derivative estimates. While this may be an acceptable choice for some applications [Mildenhall et al. 2020; Müller et al. 2021], many other situations call for different loss functions [Hasselgren et al. 2020; Heitz et al. 2021; Johnson et al. 2016; Kallweit et al. 2017; Munkberg et al. 2021; Xing et al. 2022].

[1]For the $\mathcal{L}^2$ loss specifically, there exist alternative solutions [Deng et al. 2022; Pidhorskyi et al. 2022] that not only produce unbiased gradients but also unbiased loss values (though the values do not affect optimization, see Figure 6). In our work, we argue for supporting different loss functions so we do not discuss these solutions in further detail.

[2]For convex f, this is also known as Jensen's inequality [1906].

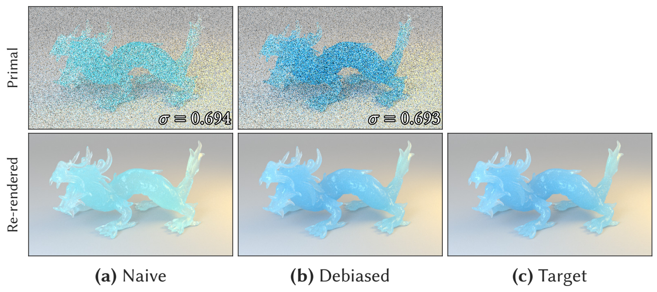


Fig. 5. Bias arising from the use of non-$\mathcal{L}^2$ losses can cause undesirable drift during the optimization. The reconstruction shown here tries to infer the medium extinction of the target **(c)** using an $\mathcal{L}^1$ loss. **(a)** Variance in the primal image biases the gradient computation, causing the optimization to stabilizes at an extinction value that is too low. **(b)** Shifting the mean (Equation (14)) to remove bias fixes the convergence. While not a practical or general solution, this emphasizes the source of the problem. The bottom row shows high sample count renderings of the final parameter states.

The issue is not just theoretical and impacts any inverse rendering task involving a non-$\mathcal{L}^2$ loss. Figure 5 demonstrates the problem in a low-dimensional reconstruction task involving the extinction of a homogeneous volume. This scene is challenging to render, causing primal images to be relatively noisy. Coupled with an $\mathcal{L}^1$ loss, the optimization stabilizes at a fixed point that does not match the target. We also perform the same optimization with a *brute force debiased* estimator of the loss derivative, in which we separately compute and subtract the bias term (we omit the I_{ref} argument for readability):

$$\langle \partial\mathcal{L}(F_\theta)\rangle_{\text{BFD}} = \partial\mathcal{L}(\langle F_\theta\rangle) + \partial\mathcal{L}(\mathbb{E}[\langle F_\theta\rangle]) - \mathbb{E}[\partial\mathcal{L}(\langle F_\theta\rangle)], \quad (14)$$

While this estimator shares the variance characteristics of the original, the lack of bias now enables the same optimization to converge to the reference answer (see Figure 5 (b)). Naturally, brute force debiasing is not a practical solution—our example merely demonstrates that the unsatisfactory convergence is indeed caused by bias.

In any application involving a non-$\mathcal{L}^2$ loss, our control variate therefore serves an important second purpose besides variance reduction, which is to reduce bias arising from the loss function. It prevents the situation where bias causes the optimization to settle in an undesirable parameter state; we visualize this situation in Figure 6. Once the optimization approaches a fixed point, our control variate reuses samples across an increasing number of iterations, improving the accuracy of the primal rendering that in turn reduces bias in the loss derivative.

Figure 7 illustrates the effectiveness of our control variate (third column). It exhibits low variance and good agreement with reference gradients computed using a high sample count. The $\mathcal{L}^1$ loss produces ±1-valued derivatives and reveals noticeable disagreement between the reference and PRB baseline.

5 IMPLEMENTATION

The control variate in Equation (6) references the terms $\langle F_n\rangle_n$ and $\langle F_{n-1}\rangle_n$, which represent correlated renderings of the scene at adjacent iterations performed using the same Monte Carlo samples. This still leaves some room for experimentation—for example, the preceding sections did not state the domain on which samples should be generated. We tested sample reuse on *primary sample space* [Kelemen et al. 2002] and on *path space* [Veach 1997]. Figure 8 visualizes the impact of this choice on the control variate's performance.

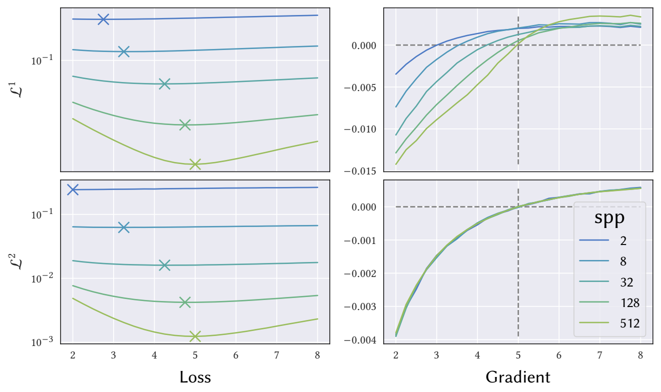


Fig. 6. 1D Visualization of the loss (left) and derivative (right) landscape of the volumetric dragon scene from Figure 5 for various extinction values $\sigma_t \in [2, 8]$ (horizontal axis), sample counts (spp), and $\mathcal{L}^1/\mathcal{L}^2$ losses (top/bottom). The reference is $\sigma_t = 5$, while loss minima (× symbols) curiously occur at different positions. This bias results from variance and the non-linearity of both losses. In the $\mathcal{L}^1$ case, bias also contaminates the derivative, causing undesirable drifts (top right). The $\mathcal{L}^2$ derivative is linear, hence the optimization has a fixed point at the correct value (bottom right).

Primary sample space. Sample reuse on primary sample space is particularly simple: all one needs to do is to render the scene twice at every iteration. The sample generator should be initialized so that its internal state in every first rendering step matches that of the preceding iteration's second rendering step. We denote this approach as "CV-PSS" in our experiments.

There are two downsides: first, rendering twice at every iteration naturally doubles the primal rendering cost, which accounts for roughly 1/3 of the total cost of a PRB gradient step.

The second issue is the way in which certain scene parameters influence the sampled path geometry. For example, consider sampling the directional distribution of a rough microfacet BRDF using a fixed position in primary sample space. Perturbing the roughness parameter will also change the computed ray direction, which can cascade into a significant change to the rest of the light path. The resulting loss of correlation is apparent in Figure 8 (b). Parameters with this behavior include surface roughness, normal maps, the index of refraction, and the extinction and scattering anisotropy of participating media.

Path space. The path-space variant reuses sampled light paths instead of the uniform variates that were used to generate them. The architecture of our implementation ("CV-PS") follows Rousselle et al. [2016], who introduce BSDF adapters that encapsulate two BSDFs with different parameter states. We extend this idea to media and rely on a modified volumetric path tracer that samples paths from the new state, while simultaneously computing radiance estimates for the old state. Participating media furthermore require that free-flight distances are sampled with a majorant that bounds both states. Appendix B provides pseudocode explaining the necessary modifications in the surface case.

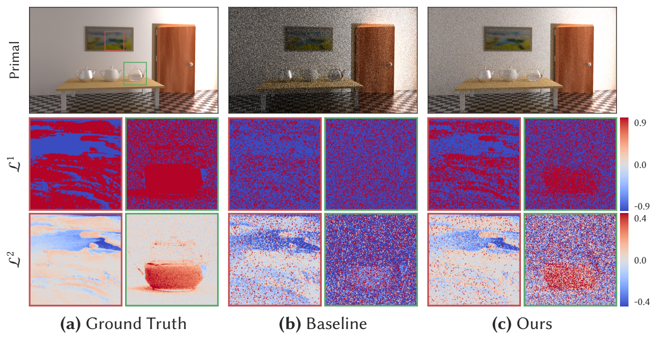


Fig. 7. Our control variate reduces both variance and bias in typical reconstruction tasks. Here, we contrast loss function derivatives of the baseline (PRB) and our method computed using either a $\mathcal{L}^1$ or $\mathcal{L}^2$ loss as loss function. While our method cannot fully remove all bias in the $\mathcal{L}^1$ case, the output still shows good agreement with reference derivatives computed using a high sample count. The example shows iteration #20 of an optimization that reconstructs the materials of a challenging interior scene lit entirely using indirect illumination, where all approaches use *the same* gradient updates, i.e. control variates are only used for visualization. Further detail on the optimization is provided in Section 6 and Figure 9.

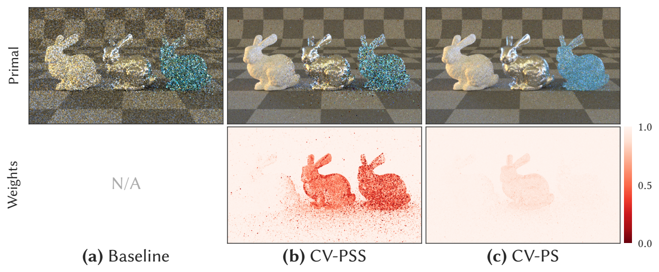


Fig. 8. Our method builds on the ability to re-render a changing scene using the same set of Monte Carlo samples. The three columns show renderings of the scene in Figure 2 at taking the same 50 optimization steps. They compare **(a)** The baseline without control variate. **(b)** Sample reuse in *primary space* ("CV-PSS"). While this approach is easy to implement by re-seeding the sample generator, it can suffer from reduced correlation when scene parameters impact the sampled path geometry. This is apparent in the optimization of bunny #2's roughness and bunny #3's extinction parameter. **(c)** Sample reuse in *path space* ("CV-PS") leads to an overall higher correlation, control weight values, and lower variance.

It is interesting to note that rendering algorithms already evaluate correlated estimators in path space, using a single path to estimate the radiance for each of several color channels. Another architectural approach for realizing path-space sample reuse could therefore entail changing the system's color representation from 3 (RGB) to 6 channels or tracking two packets of sampled wavelengths.

Path space sample reuse yields improved correlation and lower variance. Reusing light paths to evaluate two estimators also reduces ray tracing cost and makes it possible to fuse the entire computation into a single GPU kernel. Applied to the experiment in Figure 8, we find that this approach reduces primal rendering overheads from 2× to 1.6× (which, again, only represents roughly 1/3 of the total cost of a gradient step). Besides Figure 8, all other results discussed this paper use the CV-PS variant.

Geometric optimization. We did not investigate parameters that move or deform surface geometry, which pose additional challenges for sample reuse. In the primary sample space setting, we expect such parameters to reduce the correlation and effectiveness of the control variate similar to other parameters that have an effect on the sampled path geometry. The path space variant would require the addition of "shift mappings" to ensure that the moved path vertices remain valid [Manzi et al. 2016]. We find this an interesting direction for future work but consider it beyond the scope of this article.

Volumetric representations replace discrete visibility changes at boundaries with smooth transitions. They are less affected by these challenges and lend themselves to gradually evolving geometry.

6 RESULTS

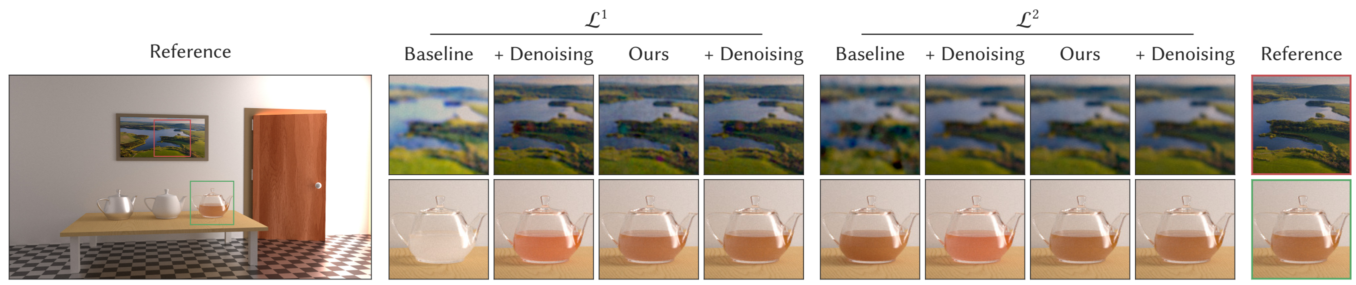
We implemented our method in Mitsuba 3 [Jakob et al. 2022b] and combined it with two distinct differentiable rendering techniques: *Path Replay Backpropagation* (PRB) [Vicini et al. 2021] to reconstruct textured surface materials and homogeneous volume parameters, and *Differential Ratio Tracking* (DRT) [Nimier-David et al. 2022] to reconstruct heterogeneous volumes. We construct a separate control weight per color channel. Figures that visualize control weights show the ones for the red channel. Our reference implementation can be found at https://github.com/rgl-epfl/recursive_control_variates.

Material reconstruction. The inversion task in Figure 9 analyzes the performance of material reconstruction in a scene with challenging indirect illumination. We hold the geometry fixed and reset the painting's texture and teapot materials. The optimization then tries to recover these properties, as well as homogeneous volume parameters for the teapot. We use the gradient preconditioning approach of [Nicolet et al. 2021] for the texture gradients. We use PRB [Vicini et al. 2021] as baseline and experimented with $\mathcal{L}^1$ and $\mathcal{L}^2$ losses that compare the tentative state to a single rendered reference image. Combining PRB with our recursive control variate results in a clear improvement irrespective of the chosen loss function. The improvement is largest when minimizing the $\mathcal{L}^1$ loss due to the bias reduction afforded by our technique.

Denoising. Most real-world use of Monte Carlo rendering nowadays involves a dedicated denoising step, which is typically carried out by convolutional neural networks trained on large datasets. It is tempting to incorporate a similar denoising step to reduce primal rendering variance in an inverse rendering pipeline.

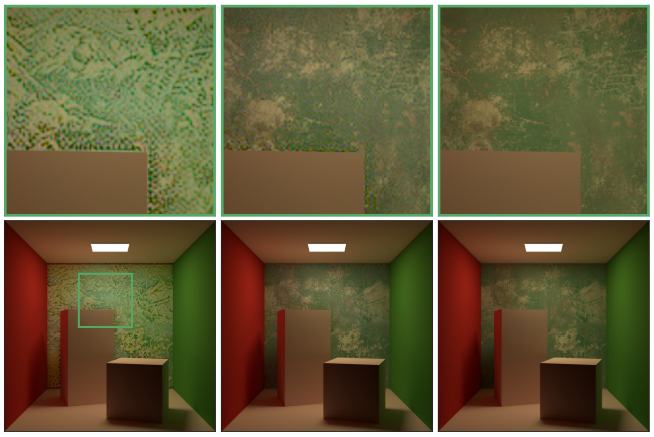
Several insets in Figure 9 demonstrate the consequences of this idea. Counter-intuitively, a direct application of a denoiser actually *reduces* the reconstruction quality, since the added bias impedes the optimization. This can turn a previously unbiased optimization into one that no longer converges— observe, e.g., the unrealistic brightening of the tea in the $\mathcal{L}^2$ PRB+Denoising column.

Our experiment used the OptiX denoiser that is based on an article by Chaitanya et al. [2017], but we expect similar behavior with other learned denoisers. Denoising our control variate estimator progressively reduces bias and yields the overall best result.



	$\mathcal{L}^1$ Baseline	+ Denoising	Ours	+ Denoising	$\mathcal{L}^2$ Baseline	+ Denoising	Ours	+ Denoising
$\mathcal{L}^1$:	2.42e-02	1.47e-02	1.45e-02	1.44e-02	1.49e-02	1.49e-02	1.46e-02	1.46e-02
$\mathcal{L}^2$:	3.05e-03	5.17e-04	4.63e-04	4.60e-04	4.97e-04	5.18e-04	4.74e-04	4.72e-04

Fig. 9. Reconstruction performance in a challenging scene that is entirely lit using indirect illumination. We reset the painting's texture and teapot materials and attempt to reconstruct them using a single rendered reference image using *Path Replay Backpropagation* [Vicini et al. 2021]. Our control variate ("Ours" columns) improves gradients, which leads to higher-quality reconstructions. The improvement consists of reduced variance when using a $\mathcal{L}^2$ loss and additional bias reduction when using a $\mathcal{L}^1$ loss and/or denoiser. Counter-intuitively, the addition of a denoising step can reduce reconstruction quality: while it lowers primal rendering noise, the added bias tends to impede the optimization. Applying the denoiser on top of our control variate estimator yields better results.



(a) Baseline **(b)** Ours **(c)** Reference

Fig. 10. Learned perceptual losses are sensitive to noise, which can bias the optimization towards suboptimal solutions. Here, we employ a VGG loss [Johnson et al. 2016] to quantify the distance to the reference **(c)** in a reconstruction of the back wall's albedo texture. The PRB baseline **(a)** is visibly affected by loss-induced bias, while our method in **(b)** yields a better match.

Neural loss. Learned neural loss functions play an increasingly important role as a differentiable proxy for perceptual similarity. VGG [Simonyan and Zisserman 2015] is a widely used convolutional neural network that has been trained to classify images from the ImageNet dataset with high accuracy. The related VGG loss [Johnson et al. 2016] truncates VGG and computes the Euclidean distance between high-level features produced during the model's evaluation.

Such techniques present an interesting alternative to traditional pixel-wise losses, but they also tend to be highly sensitive to noise due to the deep features propagating it.

The optimization in Figure 10 shows this by attempting to reconstruct the textured back wall of a Cornell box, as measured by VGG feature distance to a reference. Column (a) shows that loss-induced bias is a severe problem, while our control variate in (b) achieves a superior match to the reference in (c).

Heterogeneous volume reconstruction. Figure 11 applies our technique to the problem of heterogeneous volume reconstruction building on the adjoint pass of DRT. We modified the original implementation provided by the authors so that it processes full images instead of randomly sampled subsets, which our implementation does not currently support. We consistently use the same 8-10 reference views for each reconstruction task. Experiments in the original article [Nimier-David et al. 2022] relied on a $\mathcal{L}^1$ loss, which made it necessary to render primal images with a relatively large sample count to reduce the impact of loss function bias. Variance reduction from the control variate alleviates the need for this high sample count, which improves the reconstruction quality at equal time and significantly reduces primal rendering overheads for when targeting an equal level of quality. Detailed inspection of these results is available through our supplementary image viewer.

7 DISCUSSION

Control variates for the adjoint phase. The recursive control variate as presented in Section 3 is not limited to primal rendering—in principle, the technique applies to any sequence of Monte Carlo estimators. It should therefore be possible to also apply it to another conspicuous sequence of Monte Carlo estimators that occurs in the inverse rendering pipeline, namely the adjoint rendering phase represented by the left factor in Equation (12) that transforms the loss gradient into a scene parameter gradient.

However, correlated adjoint rendering is more involved than the sample reuse techniques described in Section 5. Furthermore, the adjoint estimator does not share the bias-related issues encountered in the primal phase. Modern optimizers such as Adam [Kingma and Ba 2015] compute accurate updates from unbiased gradients, even if they are very noisy, which makes the potential benefits uncertain. We leave further investigation of this topic as a future work.

Heuristic control weights. Our results suggest that accurate reconstruction and low variance go hand in hand with large values of the control weight α_n, e.g. in Figures 2 and 8. It is tempting to encourage such positive outcomes heuristically by setting $\alpha_n = 1$ or, informed by the steady-state solution, $\alpha_n = {}^{n}/_{n+1}$, neither of which depend on estimated statistics.

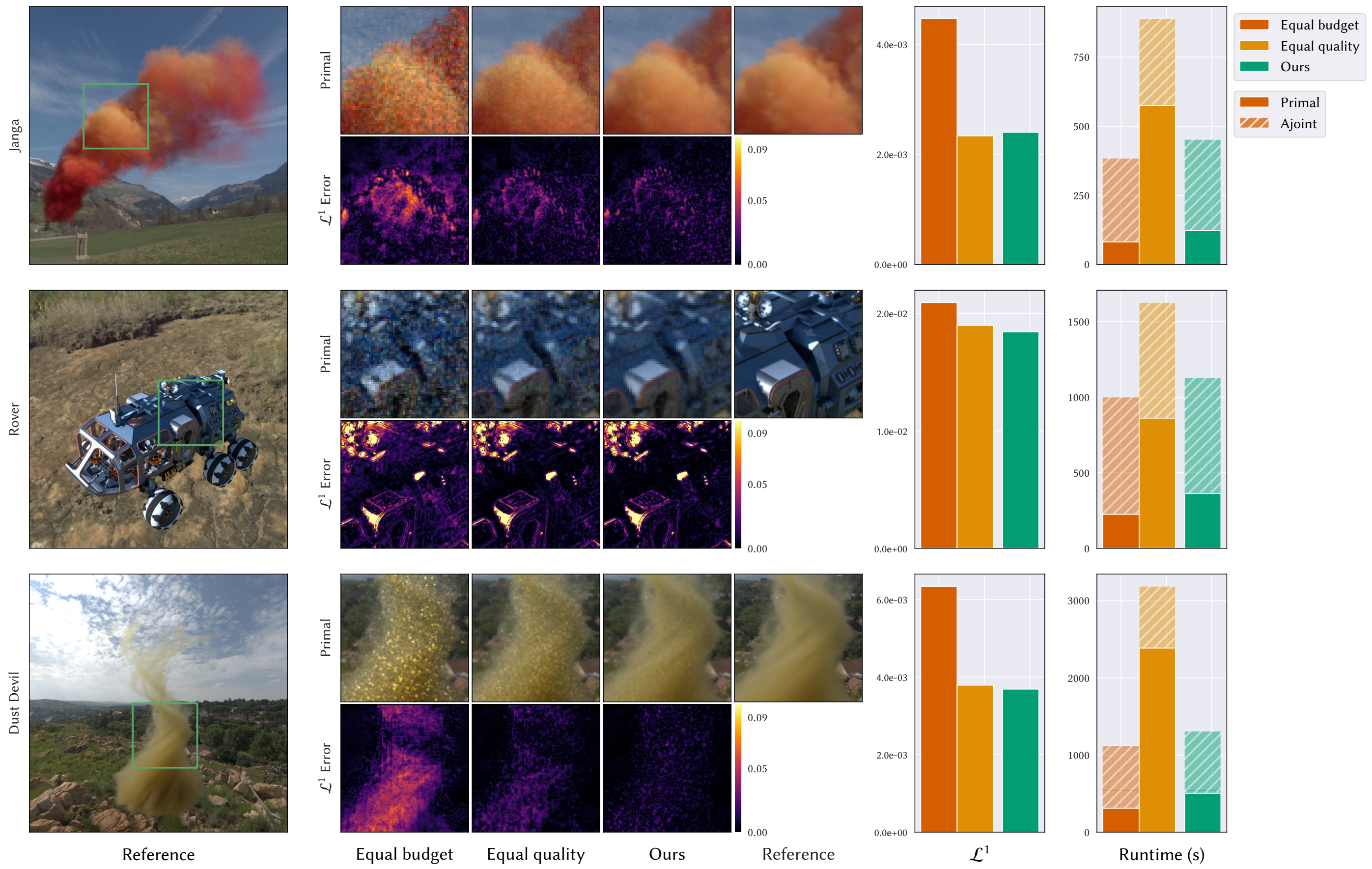


Fig. 11. We examine the effectiveness of our recursive control variate in a heterogeneous volume reconstruction based on *differential ratio tracking* (DRT) [Nimier-David et al. 2022]. The top image insets in each experiment show the primal rendering state in the last iteration of the optimization, while the bottom insets visualize the absolute difference of a high-quality re-rendered version compared to the reference. Given the same sample budget, our method achieves a higher reconstruction quality. To reach a similar level of quality, plain DRT without control variate requires ~4-8 times more samples, which comes at a significant computational cost. We only investigate changes to the primal rendering phase; the runtime cost of the DRT adjoint phase (hatched bars) is identical in every experiment, which bounds the maximum possible speedup of our method.

However, both heuristics have clear failure modes. The first results in unbounded variance because the previous control variate is never down-weighted while independent random variation continues to be added per Equation (6). The second heuristic results in a uniform combination over all prior control variates regardless of how similar each one is to the current scene. The optimal value of α_n strongly depends on the optimization trajectory, which motivates our definition of α_n in Section 3.2.

Sample reuse in volumes. Introducing correlation in heterogeneous volume rendering by reusing sampled path vertices can increase the cost of rendering. Delta tracking [Woodcock et al. 1965] requires the majorant to bound both old and new volumes, which adds overhead in the form of increased numbers of null collisions. The density change from one iteration to the next fortunately tends to be relatively small, which limits this additional cost. Similar to the observation regarding color channels in Section 5, rendering a volume with two states resembles rendering a volume with spectrally varying extinction and albedo. Thus, our method may also benefit from more sophisticated rendering techniques for spectrally varying volumes [Kutz et al. 2017; Miller et al. 2019].

Spatial reuse. Our method exploits redundancy by reusing information from prior optimization steps. Spatial reuse presents another optimization opportunity, as demonstrated in gradient-domain path tracing [Kettunen et al. 2015] and image space control variates [Keller 2001; Rousselle et al. 2016]. Curiously, recent inverse rendering techniques *decrease* spatial redundancy following the observation that processing randomly chosen pixels from all viewpoints improves convergence [Mildenhall et al. 2020; Nimier-David et al. 2022], as opposed to traditional rendering of coherent rays emanating from a single viewpoint. Future techniques targeting reuse across space and time will need to address such extra complications.

Multi-level Monte Carlo. Besides their theoretical optimality [Heinrich 1998], hierarchical Monte Carlo methods and especially telescopic sums have become a tool for debiasing estimators [Misso et al. 2022]. Hence there are two opportunities for future research: exploring the efficiency of a hierarchical approach [Keller 2001] and investigating in how far biased gradient estimators can be made at least consistent by means of multi-level Monte Carlo methods.

Combination with other variance reduction techniques. In the same spirit as the combination with denoising in Figure 9, we believe other, orthogonal, variance reduction techniques exploiting temporal, screen space, and scene space correlation, are applicable in tandem with our approach. Promising candidates are temporal path guiding [Dittebrandt et al. 2020] and reservoir resampling [Bitterli et al. 2020], which are both effective at spatio-temporal information reuse, are both unbiased, and are both computationally light weight.

8 CONCLUSION

Practically all contemporary physically based rendering techniques reuse information across simulated light paths in *some* form in order to be efficient, be that temporally or spatially. Differentiable rendering introduces an additional, as-yet untapped treasure trove of potential information reuse: the optimization loop. Our recursive control variate is a first attempt at leveraging this redundancy and we expect many more to be practical.

What makes our technique powerful is its generality. Section 3 applies to any Monte Carlo estimator that is used in conjunction with gradient descent, such as in simulations of quantum systems or reinforcement learning. It can be likened to what "momentum" is to gradient descent. Loss function bias (Section 4) is similarly a general problem that any optimization of a stochastic system must address. To this end, we take a first step by identifying the problem and alleviating the issue through variance reduction. We believe that there is great potential in developing general recipes for debiasing loss derivatives, as well as in developing further schemes for information reuse in optimization loops.

ACKNOWLEDGMENTS

The authors would like to thank the reviewers for their insightful feedback, Tizian Zeltner for helpful discussions in the early stages of the project, Valentin Deschaintre for disussions on the VGG loss, and Aaron Lefohn for his support. Volumes, environment maps and 3D models were generously provided by JangaFX, PolyHaven and vajrablue. This project has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation program (grant agreement No 948846).

REFERENCES

Dejan Azinovic, Tzu-Mao Li, Anton Kaplanyan, and Matthias Nießner. 2019. Inverse path tracing for joint material and lighting estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2447–2456.

Sai Bangaru, Tzu-Mao Li, and Frédo Durand. 2020. Unbiased Warped-Area Sampling for Differentiable Rendering. *ACM Trans. Graph.* 39, 6 (2020), 245:1–245:18.

Sai Praveen Bangaru, Michael Gharbi, Fujun Luan, Tzu-Mao Li, Kalyan Sunkavalli, Milos Hasan, Sai Bi, Zexiang Xu, Gilbert Bernstein, and Fredo Durand. 2022. Differentiable Rendering of Neural SDFs through Reparameterization. In *SIGGRAPH Asia 2022 Conference Papers*. 1–9.

Jonathan T. Barron, Ben Mildenhall, Matthew Tancik, Peter Hedman, Ricardo Martin-Brualla, and Pratul P. Srinivasan. 2021. Mip-NeRF: A Multiscale Representation for Anti-Aliasing Neural Radiance Fields. *arXiv* (2021). https://jonbarron.info/mipnerf/

Nikolaus Binder, Sascha Fricke, and Alexander Keller. 2022. Massively Parallel Path Space Filtering. In *Monte Carlo and Quasi-Monte Carlo Methods, MCQMC 2020, Oxford, United Kingdom, August 10–14*, Alexander Keller (Ed.). Springer, 149–168.

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.

Chakravarty R. Alla Chaitanya, Anton S. Kaplanyan, Christoph Schied, Marco Salvi, Aaron Lefohn, Derek Nowrouzezahrai, and Timo Aila. 2017. Interactive Reconstruction of Monte Carlo Image Sequences Using a Recurrent Denoising Autoencoder. *ACM Trans. Graph.* 36, 4 (jul 2017).

Chengqian Che, Fujun Luan, Shuang Zhao, Kavita Bala, and Ioannis Gkioulekas. 2020. Towards Learning-based Inverse Subsurface Scattering. 1–12. https://doi.org/10.1109/ICCP48838.2020.9105209

Miguel Crespo, Adrian Jarabo, and Adolfo Muñoz. 2021. Primary-space adaptive control variates using piecewise-polynomial approximations. *ACM Transactions on Graphics (TOG)* 40, 3 (2021), 1–15.

Ido Czerninski and Yoav Schechner. 2023. PARS - Path Recycling and Sorting for Efficient Cloud Tomography. *Intelligent Computing* 0, ja (2023). https://doi.org/10.34133/icomputing.0007 arXiv:https://spj.science.org/doi/pdf/10.34133/icomputing.0007

Aaron Defazio, Francis Bach, and Simon Lacoste-Julien. 2014. SAGA: A fast incremental gradient method with support for non-strongly convex composite objectives. *Advances in neural information processing systems* 27 (2014).

Xi Deng, Fujun Luan, Bruce Walter, Kavita Bala, and Steve Marschner. 2022. Reconstructing Translucent Objects using Differentiable Rendering. In *ACM SIGGRAPH 2022 Conference Proceedings*. 1–10.

Addis Dittebrandt, Johannes Hanika, and Carsten Dachsbacher. 2020. Temporal Sample Reuse for Next Event Estimation and Path Guiding for Real-Time Path Tracing. In *Eurographics Symposium on Rendering - DL-only Track*, Carsten Dachsbacher and Matt Pharr (Eds.). The Eurographics Association. https://doi.org/10.2312/sr.20201135

Oskar Elek, Tobias Ritschel, Alexander Wilkie, and Hans-Peter Seidel. 2012. Interactive Cloud Rendering Using Temporally-Coherent Photon Mapping. In *Proceedings of Graphics Interface 2012* (Toronto, Ontario, Canada) *(GI '12)*. Canadian Information Processing Society, CAN, 141–148.

Shaohua Fan, Stephen Chenney, Bo Hu, Kam-Wah Tsui, and Yu-chi Lai. 2006. Optimizing control variate estimators for rendering. In *Computer Graphics Forum*, Vol. 25. Wiley Online Library, 351–357.

Edgar C. Fieller and H. O. Hartley. 1954. Sampling with control variables. *Biometrika* 41, 3-4 (12 1954).

Iliyan Georgiev, Zackary Misso, Toshiya Hachisuka, Derek Nowrouzezahrai, Jaroslav Křivánek, and Wojciech Jarosz. 2019. Integral formulations of volumetric transmittance. *ACM Transactions on Graphics (TOG)* 38, 6 (2019), 1–17.

Ioannis Gkioulekas, Shuang Zhao, Kavita Bala, Todd Zickler, and Anat Levin. 2013. Inverse volume rendering with material dictionaries. *ACM Transactions on Graphics (TOG)* 32, 6 (2013), 1–13.

Robert M. Gower, Mark Schmidt, Francis Bach, and Peter Richtárik. 2020. Variance-reduced methods for machine learning. *Proc. IEEE* 108, 11 (2020), 1968–1983.

Jon Hasselgren, Nikolai Hofmann, and Jacob Munkberg. 2022. Shape, Light, and Material Decomposition from Images using Monte Carlo Rendering and Denoising. *arXiv:2206.03380* (2022).

Jon Hasselgren, Jacob Munkberg, Jaakko Lehtinen, Miika Aittala, and Samuli Laine. 2021. Appearance-Driven Automatic 3D Model Simplification. In *Eurographics Symposium on Rendering*.

Jon Hasselgren, Jacob Munkberg, Marco Salvi, Anjul Patney, and Aaron Lefohn. 2020. Neural Temporal Adaptive Sampling and Denoising. *Computer Graphics Forum* 39, 2 (2020), 147–155. https://doi.org/10.1111/cgf.13919 arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1111/cgf.13919

Stefan Heinrich. 1998. A Multilevel Version of the Method of Dependent Tests. In *Proc. of the 3rd St. Petersburg Workshop on Simulation*. St. Petersburg University Press, 31–35.

Eric Heitz, Kenneth Vanhoey, Thomas Chambon, and Laurent Belcour. 2021. A Sliced Wasserstein Loss for Neural Texture Synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 9412–9420.

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

Wenzel Jakob, Sébastien Speierer, Nicolas Roussel, and Delio Vicini. 2022a. Dr.Jit: A Just-In-Time Compiler for Differentiable Rendering. *Transactions on Graphics (Proceedings of SIGGRAPH)* 41, 4 (July 2022). https://doi.org/10.1145/3528223.3530099

Johan L. W. V. Jensen. 1906. Sur les fonctions convexes et les inégualités entre les valeurs Moyennes. https://doi.org/10.1007/bf02418571

Justin Johnson, Alexandre Alahi, and Li Fei-Fei. 2016. Perceptual Losses for Real-Time Style Transfer and Super-Resolution. In *Computer Vision – ECCV 2016*. Springer International Publishing.

Rie Johnson and Tong Zhang. 2013. Accelerating Stochastic Gradient Descent using Predictive Variance Reduction. In *Advances in Neural Information Processing Systems*, C.J. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K.Q. Weinberger (Eds.), Vol. 26. Curran Associates, Inc.

Simon Kallweit, Thomas Müller, Brian Mcwilliams, Markus Gross, and Jan Novák. 2017. Deep scattering: Rendering atmospheric clouds with radiance-predicting neural networks. *ACM Transactions on Graphics (TOG)* 36, 6 (2017), 1–11.

Malvin Kalos and Paula Whitlock. 1986. *Monte Carlo Methods, Volume I: Basics*. J. Wiley & Sons.

Brian Karis. 2014. High Quality Temporal Anti-Aliasing. In *ACM SIGGRAPH Courses: Advances in Real-Time Rendering in Games* (Vancouver, Canada). ACM, New York, NY, USA. https://doi.org/10.1145/2614028.2615455

Hiroharu Kato, Yoshitaka Ushiku, and Tatsuya Harada. 2017. Neural 3D Mesh Renderer. *CoRR* abs/1711.07566 (2017). arXiv:1711.07566 http://arxiv.org/abs/1711.07566

Csaba Kelemen, László Szirmay-Kalos, György Antal, and Ferenc Csonka. 2002. A Simple and Robust Mutation Strategy for the Metropolis Light Transport Algorithm. *Comput. Graph. Forum* 21, 3 (2002).

Alexander Keller. 2001. Hierarchical Monte Carlo Image Synthesis. *Mathematics and Computers in Simulation* 55, 1-3 (2001), 79–92.

Markus Kettunen, Marco Manzi, Miika Aittala, Jaakko Lehtinen, Frédo Durand, and Matthias Zwicker. 2015. Gradient-domain path tracing. *ACM Transactions on Graphics (TOG)* 34, 4 (2015), 1–13.

Pramook Khungurn, Daniel Schroeder, Shuang Zhao, Kavita Bala, and Steve Marschner. 2016. Matching Real Fabrics with Micro-Appearance Models. *ACM Trans. Graph.* 35, 1, Article 1 (dec 2016), 26 pages. https://doi.org/10.1145/2818648

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A Method for Stochastic Optimization. In *ICLR (Poster)*.

Ivo Kondapaneni, Petr Vévoda, Pascal Grittmann, Tomáš Skřivan, Philipp Slusallek, and Jaroslav Křivánek. 2019. Optimal multiple importance sampling. *ACM Transactions on Graphics (TOG)* 38, 4 (2019), 1–14.

Peter Kutz, Ralf Habel, Yining Karl Li, and Jan Novák. 2017. Spectral and decomposition tracking for rendering heterogeneous volumes. *ACM Transactions on Graphics (TOG)* 36, 4 (2017), 1–16.

Eric P. Lafortune and Yves D. Willems. 1995a. A 5D tree to reduce the variance of Monte Carlo ray tracing. In *Eurographics Workshop on Rendering Techniques*. Springer, 11–20.

Eric P. Lafortune and Yves D. Willems. 1995b. The ambient term as a variance reducing technique for Monte Carlo ray tracing. In *Photorealistic Rendering Techniques*. Springer, 168–176.

Samuli Laine, Janne Hellsten, Tero Karras, Yeongho Seol, Jaakko Lehtinen, and Timo Aila. 2020. Modular primitives for high-performance differentiable rendering. *ACM Transactions on Graphics (TOG)* 39, 6 (2020), 1–14.

Stephen S. Lavenberg, Thomas L. Moeller, and Peter D. Welch. 1982. Statistical Results on Control Variables with Application to Queueing Network Simulation. *Operations Research* 30, 1 (1982), 182–202.

Tzu-Mao Li, Miika Aittala, Frédo Durand, and Jaakko Lehtinen. 2018. Differentiable Monte Carlo Ray Tracing through Edge Sampling. *ACM Trans. Graph. (Proc. SIGGRAPH Asia)* 37, 6 (2018), 222:1–222:11.

Daqi Lin, Markus Kettunen, Benedikt Bitterli, Jacopo Pantaleoni, Cem Yuksel, and Chris Wyman. 2022. Generalized resampled importance sampling: foundations of ReSTIR. *ACM Transactions on Graphics (TOG)* 41, 4 (2022), 1–23.

Shichen Liu, Weikai Chen, Tianye Li, and Hao Li. 2019. Soft Rasterizer: Differentiable Rendering for Unsupervised Single-View Mesh Reconstruction. *CoRR* abs/1901.05567 (2019). arXiv:1901.05567 http://arxiv.org/abs/1901.05567

Matthew M. Loper and Michael J. Black. 2014. OpenDR: An approximate differentiable renderer. In *European Conference on Computer Vision*. Springer.

Zander Majercik, Thomas Müller, Alexander Keller, Derek Nowrouzezahrai, and Morgan McGuire. 2022. Dynamic Diffuse Global Illumination Resampling. *Computer Graphics Forum* 41, 1 (2022), 158–171. https://doi.org/10.1111/cgf.14427

Marco Manzi, Markus Kettunen, Frédo Durand, Matthias Zwicker, and Jaakko Lehtinen. 2016. Temporal Gradient-Domain Path Tracing. *ACM Trans. Graph.* 35, 6, Article 246 (dec 2016), 9 pages. https://doi.org/10.1145/2980179.2980256

Adam Marrs, Josef Spjut, Holger Gruen, Rahul Sathe, and Morgan McGuire. 2018. Adaptive Temporal Antialiasing. In *Proceedings of the Conference on High-Performance Graphics* (Vancouver, British Columbia, Canada) *(HPG '18)*. Association for Computing Machinery, New York, NY, USA, Article 1, 4 pages. https://doi.org/10.1145/3231578.3231579

Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. 2020. NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis. In *ECCV*.

Bailey Miller, Iliyan Georgiev, and Wojciech Jarosz. 2019. A null-scattering path integral formulation of light transport. *ACM Transactions on Graphics (TOG)* 38, 4 (2019), 1–13.

Zackary Misso, Benedikt Bitterli, Iliyan Georgiev, and Wojciech Jarosz. 2022. Unbiased and consistent rendering using biased estimators. *ACM Transactions on Graphics (TOG)* 41, 4 (2022), 1–13.

Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. 2022. Instant Neural Graphics Primitives with a Multiresolution Hash Encoding. *ACM Trans. Graph.* 41, 4, Article 102 (July 2022), 15 pages. https://doi.org/10.1145/3528223.3530127

Thomas Müller, Fabrice Rousselle, Alexander Keller, and Jan Novák. 2020. Neural control variates. *ACM Transactions on Graphics (TOG)* 39, 6 (2020), 1–19.

Thomas Müller, Fabrice Rousselle, Jan Novák, and Alexander Keller. 2021. Real-time Neural Radiance Caching for Path Tracing. *ACM Trans. Graph.* 40, 4, Article 36 (Aug. 2021), 36:1–36:16 pages. https://doi.org/10.1145/3450626.3459812

Jacob Munkberg, Jon Hasselgren, Tianchang Shen, Jun Gao, Wenzheng Chen, Alex Evans, Thomas Müller, and Sanja Fidler. 2021. Extracting Triangular 3D Models, Materials, and Lighting From Images. *arXiv:2111.12503* (2021).

Barry L. Nelson. 1990. Control variate remedies. *Operations Research* 38, 6 (1990), 974–992.

Baptiste Nicolet, Alec Jacobson, and Wenzel Jakob. 2021. Large steps in inverse rendering of geometry. *ACM Transactions on Graphics (TOG)* 40, 6 (2021), 1–13.

Merlin Nimier-David, Thomas Müller, Alexander Keller, and Wenzel Jakob. 2022. Unbiased Inverse Volume Rendering with Differential Trackers. *ACM Trans. Graph.* 41, 4, Article 44 (July 2022), 20 pages. https://doi.org/10.1145/3528223.3530073

Merlin Nimier-David, Sébastien Speierer, Benoît Ruiz, and Wenzel Jakob. 2020. Radiative Backpropagation: An Adjoint Method for Lightning-Fast Differentiable Rendering. *Transactions on Graphics (Proceedings of SIGGRAPH)* 39, 4 (July 2020). https://doi.org/10.1145/3386569.3392406

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.

Jan Novák, Andrew Selle, and Wojciech Jarosz. 2014. Residual ratio tracking for estimating attenuation in participating media. *ACM Trans. Graph.* 33, 6 (2014), 179–1.

Art B. Owen. 2013. *Monte Carlo theory, methods and examples*.

Felix Petersen, Amit H. Bermano, Oliver Deussen, and Daniel Cohen-Or. 2019. Pix2Vex: Image-to-Geometry Reconstruction using a Smooth Differentiable Renderer. *CoRR* abs/1903.11149 (2019). arXiv:1903.11149 http://arxiv.org/abs/1903.11149

Stanislav Pidhorskyi, Timur Bagautdinov, Shugao Ma, Jason Saragih, Gabriel Schwartz, Yaser Sheikh, and Tomas Simon. 2022. Depth of Field Aware Differentiable Rendering. *ACM Transactions on Graphics (TOG)* 41, 6 (2022), 1–18.

Helge Rhodin, Nadia Robertini, Christian Richardt, Hans-Peter Seidel, and Christian Theobalt. 2015. A Versatile Scene Model with Differentiable Visibility Applied to Generative Pose Estimation. In *Proceedings of ICCV 2015*.

Fabrice Rousselle, Wojciech Jarosz, and Jan Novák. 2016. Image space control variates for rendering. *ACM Transactions on Graphics (TOG)* 35, 6 (2016), 1–12.

Corentin Salaün, Adrien Gruson, Binh-Son Hua, Toshiya Hachisuka, and Gurprit Singh. 2022. Regression-based Monte Carlo integration. *ACM Transactions on Graphics (TOG)* 41, 4 (2022), 1–14.

Christoph Schied, Christoph Peters, and Carsten Dachsbacher. 2018. Gradient Estimation for Real-Time Adaptive Temporal Filtering. *Proc. ACM Comput. Graph. Interact. Tech.* 1, 2, Article 24 (aug 2018), 16 pages. https://doi.org/10.1145/3233301

Erich Schubert and Michael Gertz. 2018. Numerically stable parallel computation of (co-) variance. In *Proceedings of the 30th International Conference on Scientific and Statistical Database Management*. 1–12.

Erich Schubert, Michael Weiler, and Hans-Peter Kriegel. 2014. SigniTrend: Scalable Detection of Emerging Topics in Textual Streams by Hashed Significance Thresholds. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*. 871–880.

Erich Schubert, Michael Weiler, and Hans-Peter Kriegel. 2016. SPOTHOT: Scalable detection of geo-spatial events in large textual streams. In *Proceedings of the 28th International Conference on Scientific and Statistical Database Management*. 1–12.

Dario Seyb, Peter-Pike Sloan, Ari Silvennoinen, Michał Iwanicki, and Wojciech Jarosz. 2020. The Design and Evolution of the UberBake Light Baking System. *ACM Transactions on Graphics (Proceedings of SIGGRAPH)* 39, 4 (July 2020). https://doi.org/10.1145/3386569.3392394

Karen Simonyan and Andrew Zisserman. 2015. Very Deep Convolutional Networks for Large-Scale Image Recognition. In *International Conference on Learning Representations*.

László Szécsi, Mateu Sbert, and László Szirmay-Kalos. 2004. Combined correlated and importance sampling in direct light source computation and environment mapping. In *Computer Graphics Forum*, Vol. 23. Wiley Online Library, 585–593.
Justin Talbot, David Cline, and Parris Egbert. 2005. Importance Resampling for Global Illumination. In *Eurographics Symposium on Rendering (2005)*, Kavita Bala and Philip Dutre (Eds.). The Eurographics Association. https://doi.org/10.2312/EGWR/EGSR05/139-146
Eric Veach. 1997. *Robust Monte Carlo Methods for Light Transport Simulation.* Ph. D. Dissertation. Stanford University, Stanford, CA.
Delio Vicini, Sébastien Speierer, and Wenzel Jakob. 2021. Path Replay Backpropagation: Differentiating Light Paths using Constant Memory and Linear Time. *Transactions on Graphics (Proceedings of SIGGRAPH)* 40, 4 (Aug. 2021), 108:1–108:14. https://doi.org/10.1145/3450626.3459804
Delio Vicini, Sébastien Speierer, and Wenzel Jakob. 2022. Differentiable Signed Distance Function Rendering. *Transactions on Graphics (Proceedings of SIGGRAPH)* 41, 4 (July 2022), 125:1–125:18. https://doi.org/10.1145/3528223.3530139
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 Trans. Graph.* 37, 4, Article 124 (jul 2018), 15 pages. https://doi.org/10.1145/3197517.3201388
B. P. Welford. 1962. Note on a method for calculating corrected sums of squares and products. *Technometrics* 4, 3 (1962), 419–420.
E. Woodcock, T. Murphy, P. Hemmings, and S. Longworth. 1965. Techniques used in the GEM code for Monte Carlo neutronics calculations in reactors and other systems of complex geometry. In *Proceedings of the Conference on Applications of Computing Methods to Reactor Problems.* Argonne National Laboratory, 557.
Tiantian Xie and Marc Olano. 2021. Real-time Subsurface Control Variates: Temporally Stable Adaptive Sampling. *UMBC Student Collection* (2021).
Jiankai Xing, Fujun Luan, Ling-Qi Yan, Xuejun Hu, Houde Qian, and Kun Xu. 2022. Differentiable Rendering Using RGBXY Derivatives and Optimal Transport. *ACM Transactions on Graphics (TOG)* 41, 6 (2022), 1–13.
Alex Yu, Sara Fridovich-Keil, Matthew Tancik, Qinhong Chen, Benjamin Recht, and Angjoo Kanazawa. 2021. Plenoxels: Radiance Fields without Neural Networks. *arXiv:2112.05131* (Dec. 2021).
Tizian Zeltner, Sébastien Speierer, Iliyan Georgiev, and Wenzel Jakob. 2021. Monte Carlo Estimators for Differential Light Transport. *Transactions on Graphics (Proceedings of SIGGRAPH)* 40, 4 (Aug. 2021). https://doi.org/10.1145/3450626.3459807
Cheng Zhang, Bailey Miller, Kai Yan, Ioannis Gkioulekas, and Shuang Zhao. 2020. Path-Space Differentiable Rendering. *ACM Trans. Graph.* 39, 4 (2020), 143:1–143:19.
Cheng Zhang, Zihan Yu, and Shuang Zhao. 2021. Path-Space Differentiable Rendering of Participating Media. *ACM Trans. Graph.* 40, 4 (2021), 76:1–76:15.

A EXPONENTIALLY WEIGHTED WELFORD'S ALGORITHM

The control variate in Section 3.2 depends on variance and covariance estimates that we compute via Welford's algorithm [1962], which requires a single pass and a fixed amount of temporary memory. One additional requirement from Section 3.2 is that the influence of prior states decays according to an *exponentially weighted moving average* (EWMA). The iteration should furthermore be initialized to remove startup bias. We have not found a variant combining these three aspects in prior work and therefore describe it here.

EWMA and truncation. Given an infinite stream of observations x_n, their exponentially weighted moving average is defined as

$$\mu_n = \alpha\mu_{n-1} + (1-\alpha)x_n \tag{15}$$

$$= (1-\alpha)\sum_{k=0}^{\infty}\alpha^k x_{n-k}. \tag{16}$$

However, an infinite history is not available in practice, which means that the sum turns finite and begins with a first observation x_0:

$$\mu_n = (1-\alpha)\sum_{k=0}^{n}\alpha^k x_{n-k}. \tag{17}$$

Suppose that all elements x_n are in fact constant, i.e., $x_n = x_0$. In this case, it would be desirable that the moving average agrees with this value, but summation of the geometric progression produces

$$\mu_n = (1-\alpha^{n+1})\,x_0. \tag{18}$$

To account for the finite past window of observations, we must scale the EWMA values by $(1-\alpha^{n+1})^{-1}$. Another way of avoiding truncation error is to assign the weight 1 to the first sample by setting $\mu_0 := x_0$. This leads to the following alternative EWMA:

$$\tilde{\mu}_n = \alpha^n x_0 + (1-\alpha)\sum_{k=0}^{n-1}\alpha^k x_{n-k}\,, \tag{19}$$

which is equivalent to infinite-extent EWMA on the stream:

$$\tilde{x}_n = \begin{cases} x_0 & n \le 0, \\ x_n & n > 0. \end{cases} \tag{20}$$

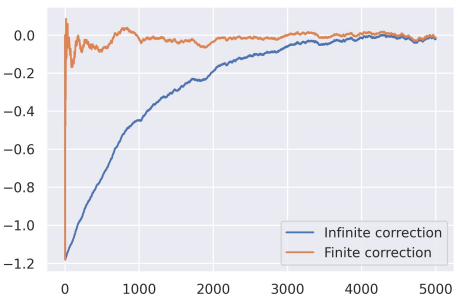


Fig. 12. Truncated EWMA of a sequence of normally distributed variates with zero mean. Assigning a weight of 1 to the first observation causes the value to linger for long after its observation, resulting in slow convergence.

Consequently, the estimator $\tilde{\mu}_n$ is unbiased when used to compute an average of iid random variables, but it its output is initially skewed towards x_0, as shown in Figure 12. This is undesirable.

```
def update_covariance(C, Mx, My, x, y, n):
    dy = y - My
    # Update the means
    My += dy / n
    Mx += (x - Mx) / n
    dx = x - Mx
    # Update the second moment
    C += dy * dx
    # Return the sample covariance
    return C / (n - 1)
```

Listing 1. Original code for Welford's online algorithm for estimating the covariance of two random variables.

Welford EWMA. We found prior use of weighted variants of Welford's algorithm work [Schubert and Gertz 2018; Schubert et al. 2014, 2016; Xie and Olano 2021], but none used the former variant of start-up bias correction described above.

The following snippet realizes a Welford EWMA for covariance estimation. Estimating the variance can be achieved in a similar manner. The example uses Python syntax somewhat leniently and assumes that an assignment to an argument propagates to the caller. We use two differential decay rates $\alpha = 0.9$ and $\beta = 0.999$ to average first and second-order statistics following Kingma and Ba [2015]. The unmodified code is provided in Listing 1.

```
def update_covariance(C, Mx, My, x, y, α, β, n):
    # Substitute the bias-corrected mean
    dy = y - My / (1 - α**(n - 1))
    # Update the means
    My = α*My + (1 - α)*y
    Mx = α*Mx + (1 - α)*x
    # Substitute the bias-corrected mean
    dx = x - Mx / (1 - α**n)
    # Update the covariance
    C = β * C + (1 - β) * dx * dy
    # Return the bias-corrected covariance
    return C / (1 - β**n)
```

Listing 2. Modified version of Welford's algorithm to use exponential moving averages to compute covariances.

B IMPLEMENTING PATH-SPACE CORRELATED RENDERING.

In order to introduce correlation in path space, one needs to evaluate two states of the scene at the same time. We achieve this by augmenting a standard path tracer by a few more function calls to evaluate the previous state of the scene while tracing. We show a simplified example for a surface path tracer below.

```
def sample(scene, ray):
    L_old, L_new = 0
    β_old, β_new = 1
    prev_bsdf_delta = True
    while(True):
        si = scene.intersect(ray)
        if si is None:
            return L_old, L_new
        # Direct emission
        if prev_bsdf_delta:
            mis_dir = mis_weight(...)
            L_old += β_old * mis_dir * si.emission
            L_new += β_new * mis_dir * si.emission
        # Emitter sampling
        wo, em_weight = scene.sample_emitter()
        val = si.bsdf.eval(wo)
        val_old = si.bsdf.eval_old(wo)
        mis_em = mis_weight(...)
        L_old += β_old * mis_em * val_old * em_weight
        L_new += β_new * mis_em * val * em_weight
        # BSDF sampling
        wo, val, pdf = si.bsdf.sample()
        # Evaluate the current path for the previous state
        val_old = si.bsdf.eval_old(wo)
        prev_bsdf_delta = si.bsdf.is_delta()
        if prev_bsdf_delta:
            β_old = 0
        else:
            β_old *= val_old / pdf
        β_new *= val / pdf
```

Listing 3. Example of the modifications for a basic surface integrator with MIS. We assume that the BSDFs have been augmented to keep track of the previous parameter state, which can be queried via `bsdf.eval_old`.