EUROGRAPHICS 2026 / B. Masia and J. Thies
(Guest Editors)

COMPUTER GRAPHICS *forum*
Volume 45 (2026), Number 2

Deep Residual Combiner: A Learned Fusion of Spatial, Temporal, and Multiscale Correlated Pixel Estimates

W. Zhou[1] and E. Hughes[1,2] and T. Hachisuka[1]

[1]University of Waterloo, Canada
[2]École de technologie supérieure, Canada

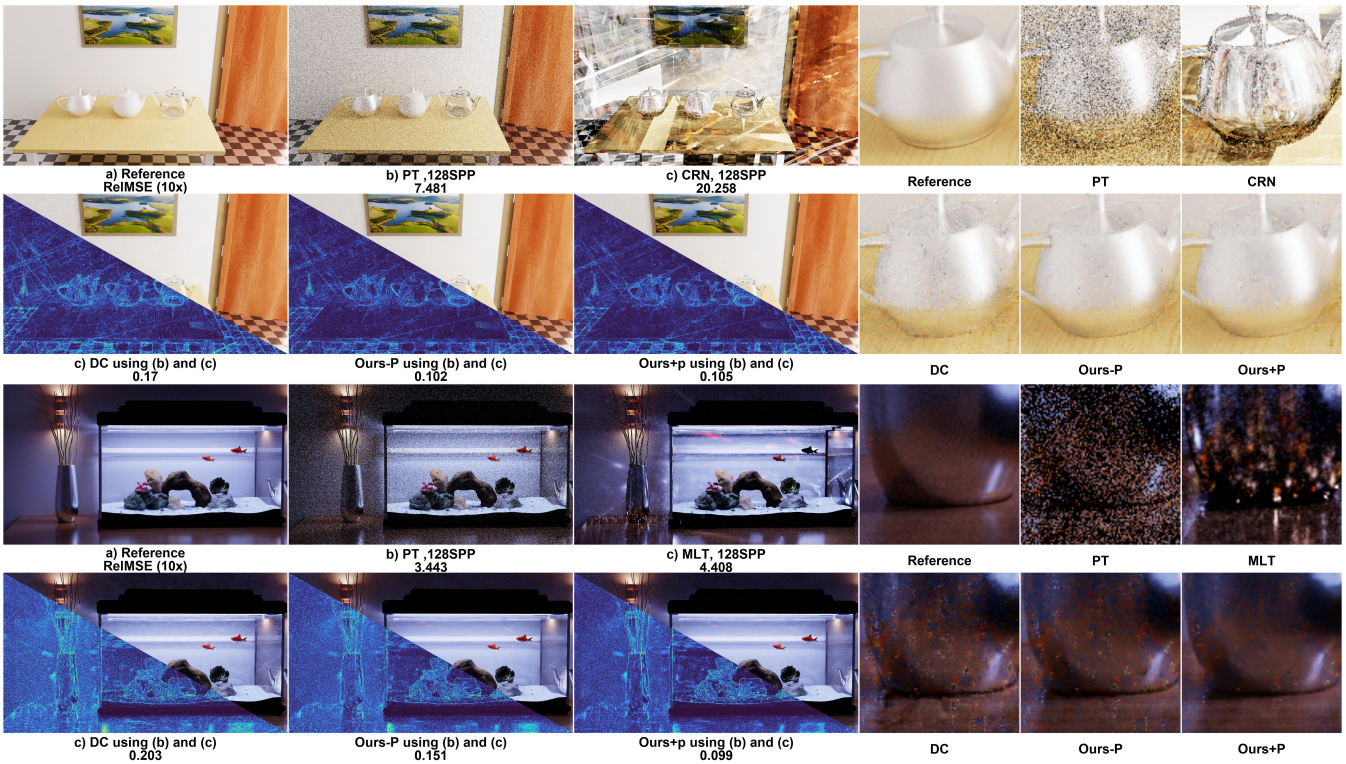

Figure 1: *Our deep residual combiner utilizes spatially and* temporally *correlated inputs (top: CRN, bottom: MLT) in addition to path tracing (PT), and combines them into a single final image. Our method benefits from having spatiotemporal correlation in those images with (Ours+P) and without (Ours-P) a new pyramidal combiner structure. Overall, our method achieves better artifact suppression and lower error compared to the previous deep combiner (DC) framework [BHHM20].*

Abstract

Correlation-based rendering techniques continue to advance, and efficiently exploiting correlations between pixel estimates has become increasingly important. The deep combiner framework [BHHM20] allows us to fuse independent and correlated pixel estimates but focuses solely on spatial correlations. We propose a generalization of the deep combiner framework, the deep residual combiner, that is designed to exploit correlations across spatial, temporal, and multiscale domains. The deep residual combiner enables robust cross-domain fusion, effectively reducing systematic artifacts and significantly enhancing temporal coherence, both of which are especially important in animation scenarios. We demonstrate the effectiveness of our proposed method through several practical applications, showcasing improvements in temporal stability, visual fidelity, and reduction of residual errors across diverse rendering scenarios compared to prior approaches.

1. Introduction

Monte Carlo rendering is the foundation of physically based rendering, providing efficient numerical solutions to complex light transport problems. The stochastic nature of Monte Carlo rendering inevitably introduces noise as numerical error in its pixel estimate, and generating noise-free images is computationally expensive. Denoising [VRM*18; MH20; BWM*23; CHL24] is one common approach to reduce noise via local filtering. While effective at reducing noise, correlations introduced by local filtering can cause structured or temporal artifacts that degrade perceptual image quality. Effectively reducing noise while avoiding the artifacts remains an open problem in rendering.

The method of control variates is a common Monte Carlo method that leverages positive correlations between samples to reduce noise. The applications of control variates go beyond physically based rendering [RJN16; MRKN20; SGH*22] and extend to inverse rendering [NRN*23], geometry processing [SC20], and fluid simulation [RSÖ*22]. Despite its effectiveness, the construction of a control variate function often incurs computational overhead, and the resulting estimator requires covariance estimation to achieve better performance beyond plain Monte Carlo methods.

Recently, the *deep combiner* (DC) [BHHM20] emerged as yet another approach to leverage correlations among pixel estimates. Unlike the method of control variates which directly improves pixel estimates, DC combines given independent and correlated pixel estimates via post-processing. Back et al. [BHHM20] recognized that denoising methods [VRM*18; MH20; BWM*23; CHL24], typically modeled as weighted sums of independent estimates among neighboring pixels, introduce *spatial* correlations among denoised pixels and thus produce correlated pixel estimates. Leveraging this insight, DC combines a noisy path-traced image with its denoised image to reduce error beyond denoising while suppressing correlation artifacts. While effective, DC focuses on processing single images, and does not explicitly consider correlation the temporal domain in animation. One can still apply DC frame-by-frame to an animated sequence, but this naive approach can introduce temporal instability, leading to perceptually disruptive artifacts such as flickering and ghosting.

We propose a generalization of the deep combiner, deep *residual* combiner, which is designed to effectively harness correlations across spatial, temporal, and even multiscale domains. Beyond incorporating spatial correlation like the deep combiner, our deep residual combiner uses bidirectional motion-based warping to reproject both past and future frames onto the current frame and thus accounts for *temporal* correlation. Inspired by recent advances in multiscale denoising [VRM*18; MH20; BWM*23; CHL24], we also introduce a pyramidal (hierarchical) combiner that captures correlations across *different resolutions* within a single image. The resulting framework is capable of fusing pixel estimates with spatiotemporal correlations from a broad range of rendering algorithms, including denoising and correlated sampling.

2. Residual Combiner

Our deep residual combiner framework builds upon the deep combiner framework [BHHM20], which introduced an approach for improving denoiser outputs by combining them with the original Monte Carlo rendering through a learned neural aggregation. Concretely, the target radiance at pixel c is estimated by blending independent estimates y (e.g., produced from Monte Carlo based rendering algorithms) with correlated estimates z (e.g., produced by denoisers) using a weighted combination kernel:

$$\hat{\beta}_c = w_c y_c + \sum_{i \in \Omega_c \setminus c} w_i \left(y_i + z_c - z_i\right), \tag{1}$$

where c denotes the center pixel, i indexes a set of its neighbors Ω_c, and the weights satisfy $\sum_{i \in \Omega_c \setminus c} w_i + w_c = 1$. The assumptions here are that estimates z_c and z_i are positively correlated and $z_c - z_i$ has a small bias. Under these assumptions, this combination can yield an improved output compared to denoising alone. While effective, DC is limited to spatial correlations, leaving its potential in other domains (e.g., temporal or multiscale) unaddressed, which are critical for temporally stable animations and preserving details at multiple resolutions. To address this, we propose a generalized framework we call the *Residual Combiner*, which extends DC by explicitly modeling residuals and accommodating correlations across spatial, temporal, and multiscale domains.

Let us first extend the notion of neighbors as follows. We adopt a unified indexing scheme: b denotes the *base* location being estimated, while r indexes a set of *residual* locations that provide correlated information. Depending on the application, r may correspond to spatial neighbors, temporal neighbors (adjacent frames), and/or multiscale neighbors across resolutions. Given a base estimator z_b of β_b, and supplementary estimators z_r and y_r of β_r, we define a two-level estimator that incorporates the residual between independent and correlated terms:

$$\hat{\beta}_b = \underbrace{z_b}_{\text{base}} + w \cdot \underbrace{(y_r - z_r)}_{\text{residual}}, \tag{2}$$

where w controls the influence of the residual term and is chosen to avoid increasing estimation error. Using identical notation (z_b, z_r) emphasizes that these estimates frequently share similar or related methodologies, resulting in positive correlations: $\mathrm{cov}(z_b, z_r) > 0$.

Typically, a zero-mean condition $E[y_r - z_r] = 0$ is assumed to avoid additional bias, but this assumption can be relaxed as discussed case-by-case. Thus, extending to multiple residuals, the generalized form becomes:

$$\hat{\beta}_b = z_b + \sum_r w_r (y_r - z_r). \tag{3}$$

This expression can be interpreted as a rearrangement of Equation 1, where $w_c y_c$ is put inside the the summation by noting $w_c y_c = w_c (y_c + z_c - z_c)$, z_c is taken outside the summation to serve as the base estimator $z_b = z_c$, and the remaining terms constitute the residual in the summation. Hence, this representation is equivalent to DC, with the relaxation that the correlated estimate z may come from a broader set of sources, including spatial neighbors, temporal neighbors, and multiscale neighbors across resolutions. A complete derivation of this equivalence is provided in Appendix B.

2.1. Error Analysis

Unlike the original deep combiner framework, our method does not require estimates (y) to be independent or unbiased. The bias of our

generalized estimator is:

$$\text{bias}(\hat{\beta}_b) \approx E[\hat{\beta}_b] - \beta_b = \text{bias}(z_b) + \sum_r w_r(\text{bias}(y_r) - \text{bias}(z_r)), \tag{4}$$

and the variance is expressed as:

$$\begin{aligned}\text{var}(\hat{\beta}_b) = \text{var}(z_b) &+ \sum_r w_r^2 \cdot \text{var}(y_r) + \sum_r w_r^2 \cdot \text{var}(z_r) \\ &+ 2\sum_r w_r \cdot \text{cov}(z_b, y_r) \\ &+ 2\sum_{s<r} w_r w_s \cdot (\text{cov}(y_r, y_s) + \text{cov}(z_r, z_s)) \\ &- 2\sum_{s<r} w_r w_s \cdot (\text{cov}(y_r, z_s) + \text{cov}(z_r, y_s)) \\ &- 2\sum_r w_r^2 \cdot \text{cov}(y_r, z_r) - 2\sum_r w_r \cdot \text{cov}(z_b, z_r)\end{aligned} \tag{5}$$

This formulation suggests that the variance can be potentially reduced by adjusting the weights for correlated estimates.

The optimal weights are those that minimize the Mean Squared Error (MSE), which should incorporate the estimator's bias, variance, and covariance. A direct calculation of the MSE and bias, however, would require the ground truth, which is unavailable in practice. Even with ground-truth data, in our setup, determining the optimal weights would require solving, for each pixel, a dense linear system with hundreds to thousands of unknowns (please see A for the full derivation of this linear system). Across an image, this amounts to on the order of one million such systems, making direct per-pixel optimization computationally prohibitive. Therefore, following DC, we employ a neural network to efficiently predict these weights. Readers interested in detailed derivations of bias, variance, and optimal weights are referred to the Appendix A.

2.2. Spatial, Temporal, and Spatiotemporal Correlation

Many rendering scenarios produce outputs that contain structured correlations. These are precisely where post-correction can provide substantial benefits.

Spatial correlation appears in applications such as denoising [BRM*16; BVM*17b] and gradient-domain rendering [KMA*15]. Our residual estimator leverages spatial correlations by aggregating residuals from neighboring pixels (Ω_c) surrounding a central pixel c:

$$\hat{\beta}_c = z_c + \sum_{i \in \Omega_c} w_i(y_i - z_i), \tag{6}$$

where weights satisfy $\sum_r w_r = 1$. Notice that this is equivalent to the DC kernel (Equation 1), but with the c term inside the sum.

Temporal correlation also presents an opportunity for correction. It naturally arises when techniques such as Common Random Numbers (CRN) are employed, where identical random numbers are reused across adjacent frames, often adopted in re-rendering applications [RJN16; XLG*24]. Our residual estimator for temporal correlation across frames centered at frame t is formulated as:

$$\hat{\beta}_t = z_t + \sum_{j \in \Gamma_t} w_j(y_j - z_j), \tag{7}$$

where Γ_t represents a temporal neighborhood around frame t.

Finally, some applications benefit most when both spatial and temporal correlations are addressed simultaneously. Examples include animation denoisers [IMF*21; BWM*23; CHL24] and spatiotemporal resampling techniques [BWP*20; LKB*22], leading to spatiotemporal correlation. For pixels around c and frames around t, our spatiotemporal residual estimator is:

$$\hat{\beta}_{c,t} = z_{c,t} + \sum_{i \in \Omega_c} \sum_{j \in \Gamma_t} w_{i,j}(y_{i,j} - z_{i,j}). \tag{8}$$

While our framework is capable of exploiting spatial correlations or temporal correlations alone, we primarily focus on the spatiotemporal case, as it is most relevant to our target applications.

2.3. Multiscale Correlations

Motivated by denoising that aggregates information from multiple resolutions, we extend our framework to exploit cross-resolution correlations. Correlations between estimates across multiple resolutions are often overlooked, yet they may present an opportunity for further error reduction. We introduce an approach employing explicit downsampling and upsampling operations. Although our framework inherently supports bidirectional correlations (i.e., both coarse-to-fine and fine-to-coarse), we focus on coarse-to-fine correlations, constructing residual estimators that propagate information from a coarser level $l-1$ to a finer level l

$$\hat{\beta}^l = z^l + w \cdot \left(U\left(y^{l-1}\right) - U\left(D\left(z^l\right)\right)\right), \tag{9}$$

where U and D denote upsampling and downsampling operators, respectively. Like many pyramidal denoising frameworks, we used a bilinear filter for D and a nearest-neighbor filter U. The term $U(D(z^l))$ downsamples z^l to the level $l-1$ and then upsamples it to the level l, so that it matches the level $l-1$ of $U(y^{l-1})$.

Directly combining estimates across scales is known to lead to undesirable ringing artifacts [BWM*23]. To mitigate this issue, we enhance our formulation by integrating spatial neighborhoods Θ_c, resulting in a spatially correlated multiscale residual estimator:

$$\hat{\beta}_c^l = z_c^l + \sum_{k \in \Theta_c} w_k \cdot \left(U\left(y^{l-1}\right)_k - U\left(D\left(z^l\right)\right)_k\right). \tag{10}$$

Given the convolutional nature of our network architecture we used for weights computation, one might expect that this multiscale approach could be learned from training data. However, as we will demonstrate later in the results, in practical scenarios where training data is not perfect, the explicit multiscale structure still serves as a useful prior to the training process.

2.4. Generalized Fusion

Our framework enables the construction of residual combiners by exploiting any combination of the correlations introduced above. We call it *fusion* as opposed to combination to emphasize the fact that correlated estimates across *different* domains are combined. This flexibility opens the door to a wide variety of application scenarios. We primarily investigate the fusion of spatiotemporal correlation (Equation 8) and multiscale correlation (Equation 10), as these two dimensions are particularly relevant for animation and re-rendering tasks. Exploring scenarios where other variants can be effectively applied remains an interesting direction for future work.

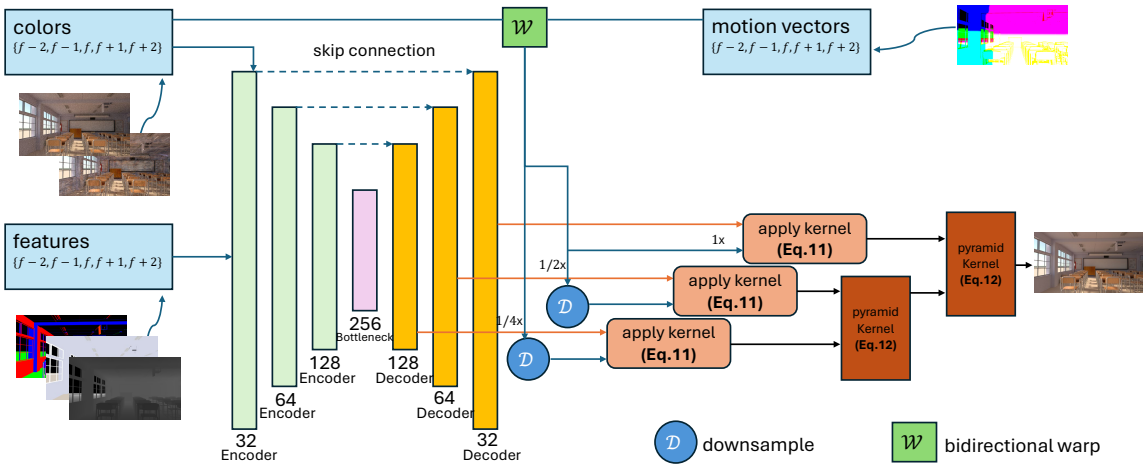

Figure 2: *The structure of our proposed network. Each decoder layer outputs a kernel for combining spatiotemporal correlated estimates (Section 3.1), and the last two layers output an extra combination kernel (Section 3.2). Each frame's input is warped as described in Section 3.1. The detailed design of each layer is presented in Fig, 3.*

3. Learning Spatial, Temporal, and Multiscale Fusion

We employ a neural network to learn fusion kernels. Our approach comprises of two stages: first, a spatiotemporal fusion of independent and correlated estimates at multiple resolutions; second, a subsequent multiscale fusion of the outputs from the initial stage.

3.1. Spatiotemporal Combiner and Bidirectional Warping

Building upon our framework, we construct a spatiotemporal residual estimator for each resolution level l defined as:

$$\bar{\beta}c,t^l = z_{c,t} + \sum_{i\in\Omega_c}\sum_{j\in\Gamma_t} w^l_{i,j}(y_{i,j} - z_{i,j}). \tag{11}$$

Our method utilizes correlations between estimates $z_{c,t}$ and $z_{i,j}$, ensuring the residual error remains small, as analyzed in Section 2.1.

To have better temporal correlation, we use a *bidirectional* strategy that considers frames symmetrically around the target frame t (from $t - \frac{n}{2}$ to $t + \frac{n}{2}$). Compared to a sequential strategy that processes only previous frames from $t - n$ to t, this bidirectional strategy can reduce inter-frame differences in general because processed frames are temporally closer to the current frame. Similar bidirectional strategies have demonstrated effectiveness across various rendering and video processing applications, including video super-resolution [CWY*21; CWX*22; HSU19], video restoration [WCY*19], and video deblurring [ZGZZ20]. Vogels et al. [VRM*18] and Kalojanov et al. [KT25] used similar approaches to achieve robust denoising results in rendering.

Beyond taking future and past frames bidirectionally in time, we also perform bidirectional motion-based warping to enhance frame alignment. Instead of directly inputting raw frames, we warp input images and associated feature buffers to the target frame t using precomputed motion vectors. The motion vectors are generated from the depth and poses, along with the camera projection and view matrix, followed by a forward-backward check to mask out the invisible pixels [ZRJ*15]. Advanced motion vector generation [ZLY*21; WZH*23; WYZ*25] is orthogonal to our framework and can further improve the performance.

For each frame j within a defined temporal window $\mathcal{T}_t$ around the frame t, we apply a warping operation $\mathcal{W}_{j\to t}(\cdot)$ that spatially aligns guide features g_j, independent estimates y_j, and correlated estimates z_j, to the frame t. We did a parameter study to investigate the impact of different warping configurations, including the directionality (bidirectional or unidirectional) and the number of frames combined. The results of this study are in Table 5. Our numerical analysis indicates that a bidirectional structure utilizing 5 combined frames yielded the most favorable performance.

3.2. Pyramidal Residual Combiner

Beyond spatiotemporal combination, we further define a pyramidal residual combiner that combines finer-resolution images with residual estimates derived from coarser resolutions. This hierarchical combination strategy allows us to achieve a wider pixel integration footprint without resorting to excessively large kernels. Our pyramidal residual combiner is formulated as follows:

$$\begin{aligned}\hat{\beta}^0_{c,t} &= \bar{\beta}^0_{c,t},\\ \hat{\beta}^l_{c,t} &= \bar{\beta}^l_{c,t} + \sum_{k\in\Theta_c} w^l_k \cdot \left(U\left(\hat{\beta}^{l-1}_t\right)_k - U\left(D\left(z^l_t\right)\right)_k\right).\end{aligned} \tag{12}$$

Note that each $\bar{\beta}^l_{c,t}$ is defined as a spatiotemporal combiner as in Equation 11. We use a compact 5×5 kernel for the pyramidal combiner, effectively balancing computational efficiency with adequate spatial support. The combined estimate at each scale is denoted by $\hat{\beta}$, while $\bar{\beta}$ refers to outputs from our spatiotemporal combiner. To ensure proper normalization, we enforce the constraint $\sum_{i\in\Omega_c}\sum_{j\in\Gamma_t} w^l_{i,j} + \sum_{k\in\Theta_c} w^l_k = 1$. This formulation allows robust fusion of spatiotemporal and multiscale information.

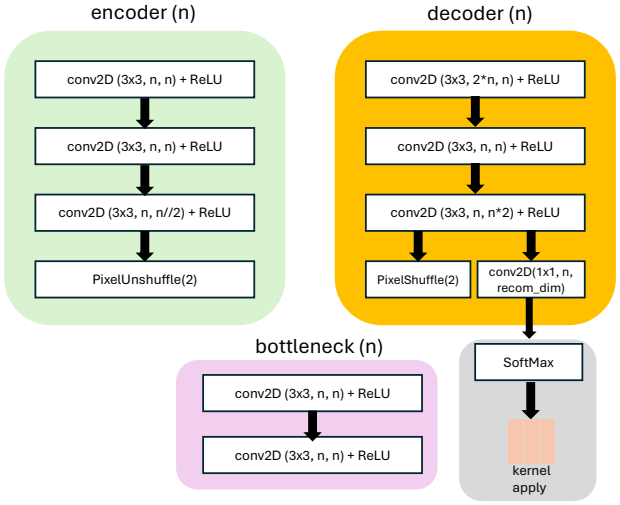

Figure 3: *Design of the encoder, bottleneck, and decoder layers used in Figure 2. In the decoder, besides passing features to upper layers, it also outputs the combination kernel, which is normalized by a softmax operation.*

3.3. Learning Weights via Neural Network

We adopt a U-Net [RFB15] architecture (illustrated in Figure 2) as our backbone due to its natural compatibility with our multiscale fusion approach. The U-Net structure effectively integrates multiscale features, capturing both global image contexts and local fine details simultaneously. Each decoding layer of the U-Net outputs spatial-temporal combination weights $w^l_{i,j}$ for resolution level l, while the final two decoding layers additionally output multiscale fusion weights w^l_k (see Figure 3).

We train our network using the Symmetric Mean Absolute Percentage Error (SMAPE) [VRM*18], defined as: $\mathrm{SMAPE}(I_{\mathrm{gt}}, I_{\mathrm{output}}) = (|I_{\mathrm{gt}} - I_{\mathrm{output}}|_1)/(|I_{\mathrm{gt}}|_1 + |I_{\mathrm{output}}|_1 + \epsilon)$, with an empirically determined parameter $\epsilon = 0.01$ to stabilize the calculation. The input to the network consists of data from n consecutive frames. Specifically, it takes the guiding features g_j (albedo, normal, and depth), independent estimates y_j, and correlated estimates z_j from frames $j \in \mathcal{T}_t$. In addition to the original guiding features in DC [BHHM20], we found that incorporating the residual signal $y_j - z_j$ as an additional guiding feature further enhances both performance and generalization. This finding may appear counterintuitive, since the network could, in principle, infer the difference between y and z directly from the two signals themselves, making the explicit residual seemingly redundant. We conjecture that the benefit arises from the limited capacity of our network (about 1.6M parameters, compared to about 20M parameters for MLTD and 30M parameters for NPPD) and the moderate size of our training dataset, where providing the residual explicitly serves as a prior. In addition, we apply a logarithmic transformation to the inputs, making the relationship between y and z non-linear and thus complicating the task of recovering their difference via training. Appendix D also provides a discussion of this behavior.

Table 1: *Quantitative comparison between baseline NPPD, NPPD+DC, and our method (wo/pyramid and w/pyramid) under equal samples (SPP). We highlight the top for each spp in orange.*

Metric	NPPD		NPPD + DC	
	8spp	32spp	8spp	32spp
PSNR ↑	29.08	31.65	28.79	31.65
SSIM ↑	0.858	0.883	0.853	0.881
ꟻLIP ↓	0.099	0.070	0.104	0.068
FoVVDP ↑	7.944	8.469	7.72	8.36
tPSNR ↑	19.49	21.16	19.26	21.06

Metric	NPPD + Ours (wo/pyramid)		NPPD + Ours (w/pyramid)	
	8spp	32spp	8spp	32spp
PSNR ↑	29.39	32.05	29.44	32.03
SSIM ↑	0.859	0.885	0.860	0.885
ꟻLIP ↓	0.090	0.060	0.086	0.060
FoVVDP ↑	7.918	8.540	8.001	8.570
tPSNR ↑	19.61	21.42	19.63	21.36

4. Application: Animation Post-Correction

Our framework incorporates spatiotemporal correlations and enables the first animation-specific post-correction method that preserves temporal stability. We evaluate two representative animation denoisers: MLTD [CHL24], which denoises MLT outputs, and NPPD [BWM*23], which denoises path-traced results. For MLTD, since it relies solely on MLT inputs, we additionally generate corresponding path-traced inputs to serve as complementary information. For NPPD, which is a path tracing denoiser, we directly combine its output with the noisy path-traced inputs. The supplemental materials contain detailed visual comparisons and video demonstrations contrasting deep combiner with our approach.

4.1. Implementation and Dataset

We implement our method using MLTD or NPPD outputs as z and the corresponding path-traced results as y in Equation 11. The comparisons against MLTD and NPPD are performed on different datasets. In particular, for NPPD, we trained our model on NPPD's publicly released dataset to enable a fair comparison. Since the complete MLTD training dataset is not publicly available, we generated new animation sequences with a modified version of the Mitsuba 0.6 renderer [Jak10] provided by the MLTD authors [Che24]. Our dataset is comprised of publicly available scenes from Bitterli's online resources [Bit16]. For each scene, we created camera trajectories to produce animated sequences and randomly extract 128×128 patches, following MLTD's dataset generation procedure. The reference images are rendered at 16k–65k spp, while training inputs are generated at 32 spp and 128 spp. For evaluation, we test on five unseen scenes: Bathroom2, Veach-Ajar, Glass-of-Water, Spaceship, and Aquarium [Bit16; RLG*20], ensuring no overlap with the training set. For NPPD, whose dataset is fully open-source, we directly use their data for both training and evaluation. For fair and consistent evaluation, we finetune the MLTD denoiser on our newly generated dataset using their released pretrained model, as detailed in Appendix C. To integrate MLTD into our framework, we also render corresponding path-traced images, which are used as an additional input channel.

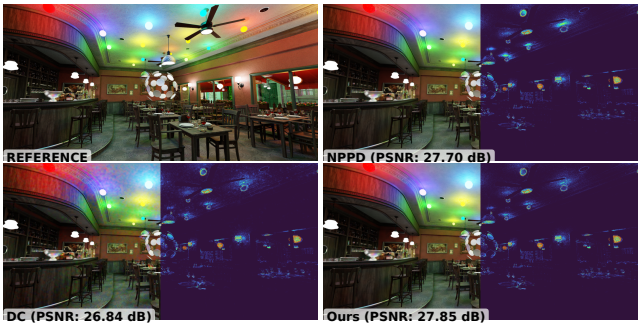

Figure 4: *Comparison of our method with the NPPD and DC baselines on the Bistro3 [Lum17] scene at 8 spp. Our approach (w/pyramid) reduces errors around the light source by leveraging path-tracing inputs. Unlike DC, which tends to introduce artifacts under low-spp conditions, our generalized fusion of multiple correlated inputs effectively suppresses such artifacts.*

4.2. Results

Table 1 presents the overall metric evaluation. We use PSNR, SSIM, and ꟻLIP [ANA*20] to assess single-frame quality, and tPSNR [MH20] with FoVVDP [MDC*21] to evaluate temporal consistency. Our method (with pyramid) consistently outperforms both the NPPD and DC baselines. While the improvements of our method may appear modest, they reflect error reductions introduced by incorporating path-traced inputs, which mostly mitigate bias (see Figure 4). When the path-tracing input is very noisy, DC can produce artifacts (Figure 4), whereas our fusion of multiple correlated inputs alleviates them. Given that our model is compact (1.6M params) compared to NPPD (30M), our approach can be employed a lightweight post-training adapter, like LoRA [HSW*21], on top of large pretrained denoisers such as NPPD.

Table 2 shows equal spp comparisons. Ours–P denotes our method without pyramid combination, while Ours+P includes the pyramid combination. Both DC and our method combine half of the samples from path tracing and half from MLTD. As shown in Table 2, our method generally outperforms both MLTD and MLTD+DC. To ensure a fair comparison, we also provide a equal runtime evaluation in Figure 8, where we use low-sample MLTD as input and compare against high-sample MLTD. Our method consistently achieves numerical improvements over MLTD by leveraging the smooth, denoised outputs of MLTD together with the high-frequency details preserved by path tracing, as demonstrated in the Glass-of-Water and Bathroom2 reflection examples.

5. Application: Spatiotemporal Correlation Combiner

Our second application focuses on combining spatiotemporally correlated estimates. This application is different from animation post correction in that the inputs are directly coming from rendering algorithms. While DC demonstrated the effectiveness of combining spatially correlated estimates using Common Random Numbers (CRN), we extend this concept further into the spatiotemporal domain along with our pyramidal combination approach. We achieve spatiotemporal correlation by ensuring that the random seed remains constant not only across pixels within each frame but throughout all pixels across an entire sequence. Beyond CRN-based input combinations, we also explore combining Metropolis Light Transport (MLT) results with path tracing outcomes, further demonstrating the broad applicability and potential of our method.

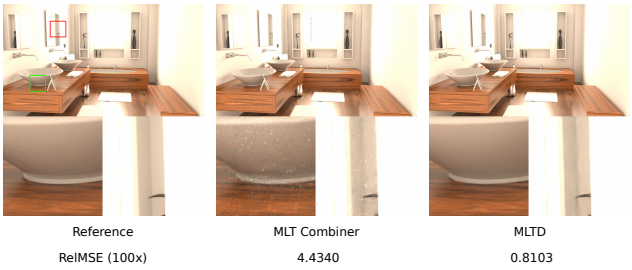

Figure 5: *The result of our combiner (combining 128spp PT and 128spp MLT) v.s. MLTD (denoising 256spp MLT).*

5.1. Implementation and Dataset

The implementation largely follows the procedure outlined in the animation post-correction application, with the key distinction that the correlated input here is specifically provided by either CRN-generated images or MLT outputs, which serve as z in Equation 11. To enable CRN generation, we extend Mitsuba 0.6 [Jak10] with explicit support for CRN rendering. All other dataset preparation steps and experimental settings remain the same, but we included the Bedroom scene [Bit16] as an extra test case.

5.2. Results

As shown in Table 2, our method, which leverages spatiotemporally correlated estimates, consistently outperforms the DC baseline. The pyramidal variant, however, tends to underperform compared to its non-pyramidal counterpart on temporal metrics, although it still yields improvements on single-frame metrics. We hypothesize that structured artifacts introduced by CRN contribute to this limitation. Importantly, this does not affect the validity of our theory, since the pyramidal combination is built on the assumption that multiscale correlation can further reduce error—an effect that is indeed observed for single-image metrics. A more detailed investigation of the conditions under which temporal performance can also be improved is left for future work. Our implementation uses a CPU-based renderer, and the runtime of our method is about 91 ms and 135 ms with and without the pyramidal combiner, compared to 36 ms for DC. We thus primarily focus on same-spp comparisons. As illustrated in Figure 1 and 7, our method consistently reduces residual error and suppresses structured artifacts that remain visible in the DC baseline, such as those in the tank and mirror in Figure 7.

We also examine the combination of MLT and path tracing using our method with a pyramidal combiner. As shown in Figure 5, our method retains noticeable noise, whereas MLTD produces smoother results. This highlights a limitation of our framework: the residual estimator (Equation 3) constrains denoising ca-

Table 2: *Comparison across sampling rates. A dedicated* Metric *column lists PSNR, SSIM, ꟻLIP, FoVVDP, and tPSNR. Dashed vertical separators distinguish subcolumns within MLTD, MLT, and CRN. Ours–P denotes the purely spatiotemporal combiner, whereas Ours+P incorporates the pyramidal combiner. We highlight the top metric using orange for each spp and for each test type (MLTD, MLT, and CRN).*

		MLTD				MLT			CRN		
SPP	**Metric**	MLTD	+DC	+Ours –P	+Ours +P	+DC	+Ours –P	+Ours +P	+DC	+Ours –P	+Ours +P
32spp	PSNR ↑	26.89	31.20	32.57	32.83	26.25	26.95	28.61	28.65	29.80	29.98
	SSIM ↑	0.879	0.895	0.909	0.913	0.626	0.697	0.799	0.794	0.837	0.842
	ꟻLIP ↓	0.126	0.095	0.078	0.071	0.126	0.113	0.106	0.109	0.101	0.100
	FoVVDP ↑	6.213	7.381	7.753	7.807	6.704	7.129	7.237	7.150	7.345	7.355
	tPSNR ↑	19.01	20.99	22.27	22.47	16.14	17.61	18.31	18.67	19.94	20.04
64spp	PSNR ↑	28.07	32.91	34.07	34.18	27.99	28.88	30.49	29.85	31.27	31.45
	SSIM ↑	0.890	0.912	0.922	0.924	0.693	0.769	0.849	0.817	0.868	0.874
	ꟻLIP ↓	0.109	0.076	0.0631	0.059	0.093	0.090	0.084	0.091	0.082	0.080
	FoVVDP ↑	6.481	7.867	8.188	8.188	7.272	7.594	7.729	7.578	7.832	7.823
	tPSNR ↑	20.48	22.46	23.48	23.52	17.64	18.98	19.91	19.65	21.15	21.14
128spp	PSNR ↑	28.98	34.38	35.30	35.31	29.62	30.69	32.16	31.13	32.64	32.68
	SSIM ↑	0.898	0.924	0.931	0.931	0.752	0.826	0.883	0.836	0.888	0.893
	ꟻLIP ↓	0.098	0.062	0.052	0.050	0.087	0.074	0.069	0.077	0.068	0.068
	FoVVDP ↑	6.676	8.262	8.275	8.496	7.747	8.027	8.115	7.942	8.226	8.195
	tPSNR ↑	21.01	23.75	24.73	24.44	19.11	20.40	21.47	20.77	22.33	22.24
256spp	PSNR ↑	29.31	35.592	34.98	36.27	31.17	32.36	33.66	32.45	34.03	34.04
	SSIM ↑	0.910	0.932	0.936	0.936	0.802	0.865	0.904	0.859	0.909	0.911
	ꟻLIP ↓	0.090	0.051	0.045	0.044	0.074	0.062	0.058	0.065	0.057	0.057
	FoVVDP ↑	7.255	8.579	8.786	8.752	8.165	8.402	8.468	8.323	8.570	8.566
	tPSNR ↑	20.54	24.77	25.27	25.23	20.60	21.83	22.90	22.07	23.69	23.62

pability compared to traditional approaches that directly average neighboring pixels. Despite the higher variance, our method better preserves fine details. For example, the mirror reflection appears aliased in MLTD but remains intact with our combiner. Furthermore, as shown in Figure 6, our approach achieves significantly lower squared bias, indicating that developing less biased denoisers within our framework is a promising direction for future work.

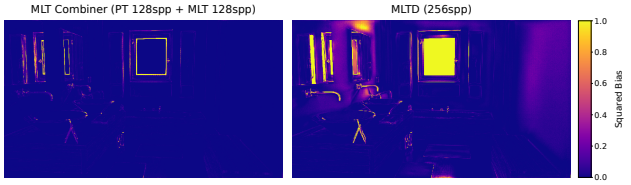

Figure 6: *Squared bias of left: our combiner (combining 128spp PT and 128spp MLT) and right: MLTD (denoising 256spp MLT). Our combiner has a significantly lower bias than MLTD, despite its higher RelMSE.*

6. Related Work

6.1. Denoising

Denoising has seen significant progress with the development of deep learning, which in many cases now surpasses plain Monte Carlo rendering alone. The first notable machine learning based denoiser for Monte Carlo rendering was introduced by Kalantari et al. [KBS15], which is followed by a wide range of neural approaches including recurrent networks [CKS*17], kernel prediction [BVM*17a], GANs [XZW*19], attention mechanisms [YNL*21], and transformer architectures [CHL24]. For a comprehensive overview of classical filtering and reconstruction techniques, we refer readers to the survey by Zwicker et al [ZJL*15].

In denoising, temporal coherence, crucial for animation, has been addressed by recurrent networks [CKS*17] and more recently recurrent transformer models [CHL24]. Our framework can be combined with these efforts by incorporating their outputs into a general formulation that further strengthens spatiotemporal coherence. In interactive rendering, strict real-time constraints often limit the exploitation of temporal data. For instance, the number of past frames kept can be limited and usually the future frames are not available. Interactive denoisers [SKW*17; MZV*20; MH20] thus cannot fully utilize temporal information,

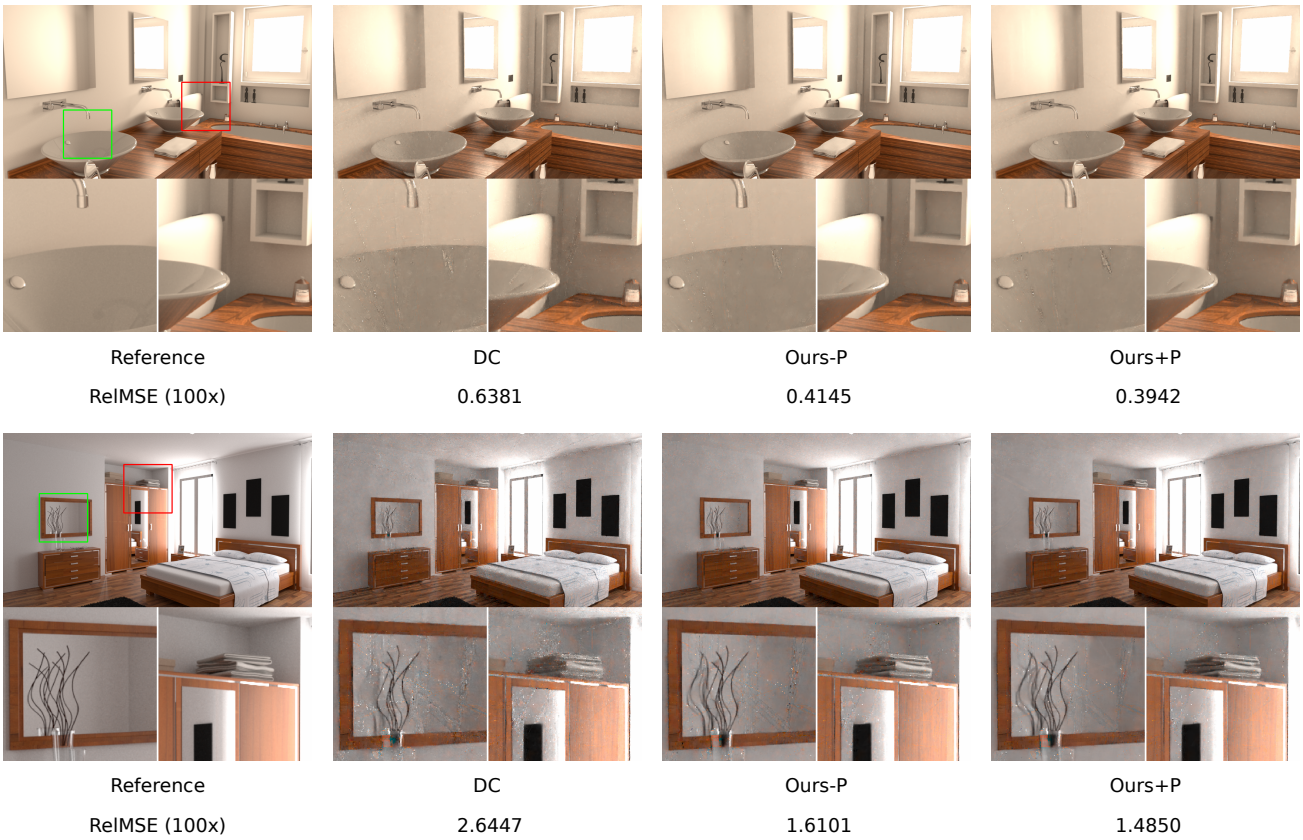

Figure 7: *CRN results for Bathroom2 (top) and Bedroom (bottom). Our method outperforms the DC baseline by better suppressing artifacts, such as those in the tank (top) and the mirror (bottom). Our–P denotes our method without pyramid combination, while Our+P incorporates pyramid combination. The pyramid combination further reduces error. The images for Bathroom2 (top) are shown with adjusted exposure for improved visualization.*

although recent work explores temporal warping with learned blending factors [IMF*21] and neural enhancements for offline animation denoising [BWM*23]. By contrast, our method employs a bidirectional *temporal kernel* that leverages both past and future frames, ensuring stronger coherence without the latency restrictions imposed by interactive settings. Our approach is related to multi-frame methods such as Vogels et al. [VRM*18], which directly integrate multiple frames into a denoising network. However, their design focuses solely on independent estimates, whereas our framework operates as a post-processing stage that combines both independent and correlated estimates, leading to improved efficiency and robustness.

Multiscale filtering can effectively suppress low-frequency noise [DMB*14; VRM*18; BWM*23; CHL24]. Inspired by this line of work, we design a multiscale module not only to separate frequencies, but to explicitly exploit correlations across scales for error reduction. Furthermore, unlike prior methods that blend results on a per-pixel basis [VRM*18; CHL24] or a splatting form [BWM*23], our framework combines images of different resolutions using a dedicated gathering kernel, yielding more stable outputs.

6.2. Post-Correction

No single denoising method consistently performs best across all scenes. The performance of deep learning–based denoisers, in particular, strongly depend on training data. To address these issues, *post-correction* techniques refine initial denoised outputs to further correct any remaining artifacts and noise. Back et al. [BHHM20] pioneered the idea of combining denoised results with raw Monte Carlo path-tracing outputs, exploiting inter-pixel correlations to suppress artifacts. Building on this, Gu et al. [GIM22] introduced James–Stein estimation to enforce statistical consistency, while Back et al. [BHHM23] leveraged the Hogg's theorem to explicitly suppress bias. Our approach extends this line of work by explicitly incorporating *spatial*, *temporal*, and *multiscale* correlations, which are particularly beneficial in animation scenarios.

Other approaches such as self-supervised correction [BHHM22; CHH*24], combination with error estimation [FRRW24], construction of consistent denoisers [FFJ22; WQH*24], and combination with adaptive sampling [FFJ23] all present complementary research avenues. Unlike these approaches, our focus is not on bias correction or consistency guarantees, but rather on exploiting richer correlation structures to improve post-correction robustness.

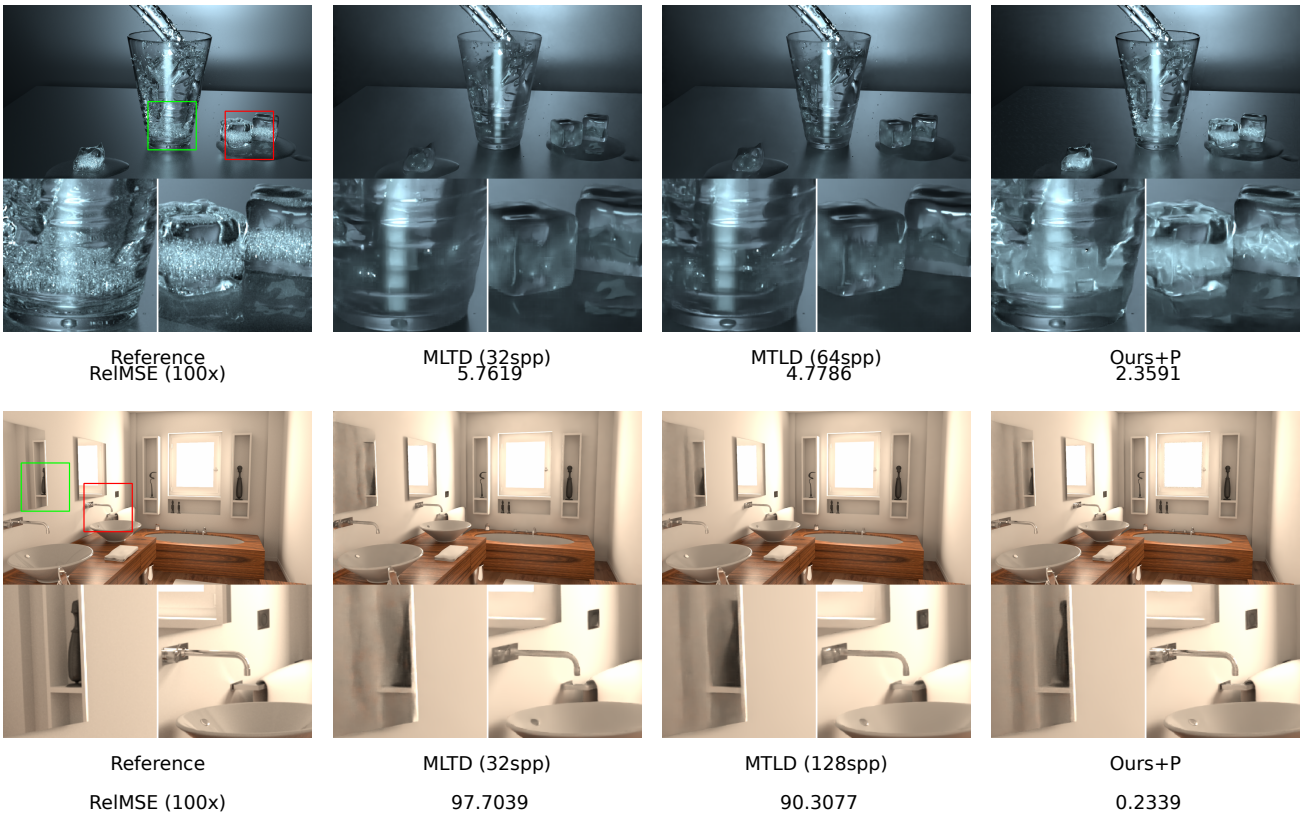

Figure 8: *Equal runtime comparison of MLTD and our method. Our+P denotes our method with pyramid combination. Our approach takes low-sample MLTD outputs (32 spp in this example) together with additional path-tracing input, achieving better quality than MLTD even at higher spp. MLTD tends to over-blur the glass (top), whereas our method partially recovers its structure. For the mirror reflection (bottom), our method successfully restores the details. Rendering times are comparable: top row — MLTD (64 spp) 9 s vs. ours 10 s; bottom row — MLTD (128 spp) 23 s vs. ours 24 s. The images for Bathroom2 (bottom) are shown with adjusted exposure for improved visualization.*

Nonetheless, integrating principles such as James–Stein estimation or the Hogg's theorem likely offers promising future directions.

6.3. Correlated Estimates

Our framework combines correlated inputs (such as CRN or MLT) with independent inputs (e.g., path tracing) to mitigate artifacts caused by correlation and enhance overall image quality. Extending this approach to an even larger family of correlated sampling methods is possible. For instance, we introduced correlation in MLT inputs by reusing seeds. However, temporal correlations remain weak in the resulting sampled paths. Joran et al. [VFBD17] strengthened spatiotemporal correlation through temporal-domain mutations. Likewise, path reuse techniques [BSH02; KDB16; WGGH20; DHC*21; WGH22] could potentially serve as correlated inputs in our framework. Extending these methods to incorporate temporal correlation, and applying our combiner framework to their outputs, is a potential avenue for future research. Similarly, ReSTIR and its variants [BWP*20; LKB*22; ZLK*24; HKL*25; OLK*21] leverage spatiotemporal reuse but suffer from correlation artifacts [SLK*24]. While these methods typically target real-time applications, adapting our framework to meet real-time constraints is a feasible direction of future work. Furthermore, the deep combiner [BHHM20] has shown compatibility with gradient-domain rendering techniques[LKL*13; KMA*15], indicating that our spatiotemporal extensions might also benefit temporal gradient-domain approaches [MKD*16].

7. Conclusion

The deep residual combiner framework enables the fusion of spatial, temporal, and multiscale correlated pixel estimates. This general framework supports a variety of Monte Carlo rendering applications, including animation post-correction and spatiotemporal estimate combination. Leveraging spatiotemporal correlation, we present what is, to our knowledge, the first animation-specific post-correction method that improves both image quality and temporal coherence over standard denoisers. In the context of combining correlated estimates, our framework outperforms the prior deep combiner that operates only on spatially independent estimates. We demonstrated the framework's flexibility by applying it to combine Metropolis light transport and path tracing results.

As shown in our results, this design leads to lower squared bias and better structural preservation (e.g., reflections). Recent work by Back et al. [BHHM23] proposes an analytical approach to bias reduction, but lacks the scalability and learning capacity of neural methods. Combining our learning-based framework with such bias-aware principles offers a promising future direction toward building unbiased, temporally coherent denoisers. We believe it opens new possibilities for exploiting correlated estimates across a wide range of Monte Carlo rendering problems.

Acknowledgement

We appreciate the anonymous reviewers for their constructive comments. We would like to thank Benedikt Bitterli for public scenes [Bit16] and the authors of NPPD [BWM*23] and MLTD [CHL24] for making their codes and pretrained model available, which was used for comparison in this work. This research was supported by NSERC under grant number RGPIN-2020-03918.

A. Error Analysis

A.1. Bias

The expectation of our combined residual estimator (Eq. 3) is:

$$\begin{aligned} E[\hat{\beta}_b] &= E\left[z_b + \sum_r w_r (y_r - z_r)\right] \\ &\approx E[z_b] + \sum_r w_r E[y_r - z_r] \\ &= \beta_b + \text{bias}(z_b) + \sum_r w_r (\beta_r - \beta_r + \text{bias}(y_r) - \text{bias}(z_r)) \\ &= \beta_b + \text{bias}(z_b) + \sum_r w_r (\text{bias}(y_r) - \text{bias}(z_r)) \end{aligned} \tag{13}$$

The approximation in the second line assumes constant weights w_r, although, strictly speaking, these weights depend on input estimates y and z. Thus, the estimator's bias is:

$$\text{bias}(\hat{\beta}_b) = E[\hat{\beta}_b] - \beta_b = \text{bias}(z_b) + \sum_r w_r (\text{bias}(y_r) - \text{bias}(z_r)) \tag{14}$$

A.2. Variance

Assuming constant weights w_r, the variance of our combined estimator is given by:

$$\begin{aligned} \text{var}(\hat{\beta}_b) &= \text{var}\left(z_b + \sum_r w_r (y_r - z_r)\right) \\ &\approx \text{var}(z_b) + \sum_r w_r^2 \text{var}(y_r - z_r) + 2\sum_r w_r \text{cov}(z_b, y_r - z_r) \\ &\quad + 2\sum_{s<r} w_r w_s \text{cov}(y_r - z_r, y_s - z_s) \end{aligned} \tag{15}$$

Applying standard variance–covariance identities, we have:

$$\begin{aligned} \text{var}(y_r - z_r) &= \text{var}(y_r) + \text{var}(z_r) - 2\text{cov}(y_r, z_r), \\ \text{cov}(z_b, y_r - z_r) &= \text{cov}(z_b, y_r) - \text{cov}(z_b, z_r), \end{aligned}$$

which leads to:

$$\begin{aligned} \text{var}(\hat{\beta}_b) &\approx \text{var}(z_b) + \sum_r w_r^2 [\text{var}(y_r) + \text{var}(z_r) - 2\text{cov}(y_r, z_r)] \\ &\quad + 2\sum_r w_r [\text{cov}(z_b, y_r) - \text{cov}(z_b, z_r)] \\ &\quad + 2\sum_{s<r} w_r w_s \text{cov}(y_r - z_r, y_s - z_s) \\ &= \text{var}(z_b) + \sum_r w_r^2 \cdot \text{var}(y_r) + \sum_r w_r^2 \cdot \text{var}(z_r) \\ &\quad + 2\sum_r w_r \cdot \text{cov}(z_b, y_r) + 2\sum_{s<r} w_r w_s \cdot \text{cov}(y_r - z_r, y_s - z_s) \\ &\quad - 2\sum_r w_r^2 \cdot \text{cov}(y_r, z_r) - 2\sum_r w_r \cdot \text{cov}(z_b, z_r) \end{aligned} \tag{16}$$

Expanding the cross-covariance term $\text{cov}(y_r - z_r, y_s - z_s)$ gives:

$$\begin{aligned} \text{cov}(y_r - z_r, y_s - z_s) &= \text{cov}(y_r, y_s) + \text{cov}(z_r, z_s) \\ &\quad - \text{cov}(y_r, z_s) - \text{cov}(z_r, y_s) \end{aligned} \tag{17}$$

Therefore, the final variance expression can be written as:

$$\begin{aligned} \text{var}(\hat{\beta}_b) &= \text{var}(z_b) + \sum_r w_r^2 \cdot \text{var}(y_r) + \sum_r w_r^2 \cdot \text{var}(z_r) \\ &\quad + 2\sum_r w_r \cdot \text{cov}(z_b, y_r) \\ &\quad + 2\sum_{s<r} w_r w_s \cdot (\text{cov}(y_r, y_s) + \text{cov}(z_r, z_s)) \\ &\quad - 2\sum_{s<r} w_r w_s \cdot (\text{cov}(y_r, z_s) + \text{cov}(z_r, y_s)) \\ &\quad - 2\sum_r w_r^2 \cdot \text{cov}(y_r, z_r) - 2\sum_r w_r \cdot \text{cov}(z_b, z_r) \end{aligned} \tag{18}$$

If we impose the assumption from DC [BHHM20] that the y are independent of z, this expression reduces to the variance form of DC (with reordering terms):

$$\begin{aligned} \text{var}(\hat{\beta}_b)_{\text{ind}} &= \text{var}(z_b) + \sum_r w_r^2 \cdot \text{var}(y_r) + \sum_r w_r^2 \cdot \text{var}(z_r) \\ &\quad + 2\sum_{s<r} w_r w_s \cdot \text{cov}(z_r, z_s) \\ &\quad - 2\sum_r w_r \cdot \text{cov}(z_b, z_r) \end{aligned} \tag{19}$$

In practice, however, this independence assumption rarely holds, since z is typically the denoiser output conditioned on y, making y and z correlated. Even so, relaxing this condition does not necessarily worsen variance, because

$$\begin{aligned} \text{var}(\hat{\beta}_b) - \text{var}(\hat{\beta}_b)_{\text{ind}} &= 2\sum_r w_r \cdot \text{cov}(z_b, y_r) \\ &\quad + 2\sum_{s<r} w_r w_s \cdot \text{cov}(y_r, y_s) \\ &\quad - 2\sum_{s<r} w_r w_s \cdot (\text{cov}(y_r, z_s) + \text{cov}(z_r, y_s)) \\ &\quad - 2\sum_r w_r^2 \cdot \text{cov}(y_r, z_r) \end{aligned} \tag{20}$$

which contains no terms that guarantee an increase in variance.

A.3. Optimal Weight Derivation

To find optimal weights minimizing mean squared error (MSE):

$$\text{MSE}(\hat{\beta}_b) = \text{bias}(\hat{\beta}_b)^2 + \text{var}(\hat{\beta}_b), \tag{21}$$

we apply the Lagrangian method under constraint $\sum_r w_r = 1$:

$$\mathcal{L}(w_r, \lambda) = \mathrm{MSE}(\hat{\beta}_b) + \lambda \left(1 - \sum_r w_r\right) \tag{22}$$

subject to the conditions:

$$\begin{aligned} \frac{\partial \mathcal{L}}{\partial w_r} &= 0, \\ \frac{\partial \mathcal{L}}{\partial \lambda} &= 1 - \sum_r w_r = 0. \end{aligned} \tag{23}$$

However, accurate MSE estimation requires ground truth, which is unavailable in practice. Furthermore, solving this optimization per pixel involves $|r|+1$ equations, making it computationally impractical, especially for our spatial, temporal, and multiscale applications where the size: $|r| > 1000$. Therefore, inspired by deep combiner theory [BHHM20], we use a neural network to predict optimal weights efficiently.

B. Equivalence of Residual Combiner and Deep Combiner

We show that our residual combiner (Equation 6 with spatial correlation) can be rewritten in the same form as the deep combiner (Equation 1). Starting from

$$\begin{aligned} \hat{\beta}_c &= z_c + \sum_{i \in \Omega_c} w_i (y_i - z_i) \\ &= z_c + \sum_{i \in \Omega_c} w_i y_i - \sum_{i \in \Omega_c} w_i z_i \\ &= \sum_{i \in \Omega_c} w_i y_i + \sum_{i \in \Omega_c} w_i z_c - \sum_{i \in \Omega_c} w_i z_i \\ &= \sum_{i \in \Omega_c} w_i (y_i + z_c - z_i), \end{aligned}$$

where the third line follows from $\sum_{i \in \Omega_c} w_i = 1$, allowing us to write $z_c = \sum_{i \in \Omega_c} w_i z_c$. Separating the center index c yields

$$\begin{aligned} \hat{\beta}_c &= w_c (y_c + z_c - z_c) + \sum_{i \in \Omega_c \setminus \{c\}} w_i (y_i + z_c - z_i) \\ &= w_c y_c + \sum_{i \in \Omega_c \setminus \{c\}} w_i (y_i + z_c - z_i), \end{aligned}$$

which matches the deep combiner form.

C. Baseline Justification for MLTD

Compared to the large-scale dataset used in MLTD [CHL24], our dataset is relatively small. This naturally raises the question of whether such a dataset is sufficient to train a network with nearly 20M parameters. To address this concern, we conduct two training strategies for MLTD: 1) retraining the model from scratch using our dataset, and 2) finetuning the provided pretrained model using our dataset.

From Table 3, it is evident that the finetuned model consistently outperforms the model retrained from scratch across all metrics and sampling rates. Consequently, we adopt the finetuned model as our baseline for MLTD. For simplicity, we denote this finetuned model as *MLTD* in the main text.

It is worth noting that while the finetuned model achieves performance comparable to the original MLTD in terms of static image metrics (SSIM, PSNR, RelMSE, and ꟻLIP), there remains a noticeable gap in temporal metrics (tPSNR and FoVVDP). We conjecture that this discrepancy arises from differences in dataset motion characteristics: our dataset exhibits faster camera movements compared to those in MLTD's dataset, leading to a mismatch in temporal stability. Importantly, this observation underscores the value of our bidirectional warping design, which leverages complementary temporal information from both past and future frames to mitigate such challenges.

Metric	16 spp		32 spp		64 spp		128 spp	
	Finetune	Retrain	Finetune	Retrain	Finetune	Retrain	Finetune	Retrain
SSIM ↑	85.62	82.62	86.92	84.22	88.05	85.56	88.93	86.53
PSNR ↑	25.28	23.57	26.50	24.87	27.64	25.99	28.54	26.99
RelMSE ↓	0.061	0.286	0.040	0.250	0.036	0.180	0.028	0.171
ꟻLIP ↓	0.149	0.163	0.132	0.143	0.114	0.129	0.103	0.115
tPSNR ↑	19.36	18.40	19.92	18.91	20.56	19.41	21.11	19.91
FoVVDP ↑	6.03	5.81	6.29	6.07	6.55	6.32	6.74	6.54

Table 3: *Comparison of metrics across different SPP settings for finetuned and retrained MLTD models. We highlight the top metric for each spp using orange.*

Metric	w/ Residual	w/o Residual
PSNR ↑	29.86	29.56
SSIM ↑	0.863	0.863
ꟻLIP ↓	0.084	0.085
FoVVDP ↑	8.190	8.127
tPSNR ↑	20.56	20.48

Table 4: *Ablation study on the residual signal $y - z$. Incorporating the residual signal yields consistent improvements across different metrics. We highlight the top metric using orange.*

D. Ablation Study on the Residual Signal

To evaluate the contribution of the residual signal $y - z$, we conduct an ablation study using NPPD as the baseline. Specifically, we train our model both with and without the residual signal while disabling the pyramid combiner, since the pyramid component does not rely on $y - z$. Table 4 summarizes the quantitative results across multiple metrics. Incorporating the residual signal consistently improves performance, albeit modestly, which indicates that it provides complementary information beyond the baseline.

These results demonstrate consistent gains over the variant without the residual signal. Based on this observation, we include the residual signal as an additional input in our model.

References

[ANA*20] ANDERSSON, PONTUS, NILSSON, JIM, AKENINE-MÖLLER, TOMAS, et al. "ꟻLIP: A Difference Evaluator for Alternating Images". *Proceedings of the ACM on Computer Graphics and Interactive Techniques* 3.2 (2020), 15:1–15:23. DOI: 10.1145/3406183 6.

Table 5: *A study of parameters on designs leveraging spatiotemporal correlation using nppd [BWM*23] as baseline. We compare four variants: 7-frame bidirectional, 3-frame bidirectional, 5-frame bidirectional (ours), and 5-frame unidirectional. The top two results for each metric are highlighted in orange and light orange, respectively. Although the 7-frame bidirectional design slightly outperforms our 5-frame bidirectional variant, the additional computational cost outweighs the marginal performance gain. Thus, we adopt the 5-frame bidirectional design for its balanced efficiency and effectiveness.*

Metric	7 frames		3 frames		5 frames		5 frames (uni)	
	8 spp	32 spp	8 spp	32 spp	8 spp	32 spp	8 spp	32 spp
PSNR↑	29.39	32.09	29.26	31.98	29.39	32.05	29.14	31.82
SSIM↑	0.859	0.886	0.857	0.884	0.859	0.885	0.856	0.883
ꟻLIP ↓	0.088	0.059	0.093	0.062	0.090	0.060	0.095	0.064
FoVVDP↑	7.954	8.564	7.834	8.480	7.918	8.541	7.807	8.434
tPSNR↑	19.61	21.45	19.53	21.36	19.61	21.42	19.46	21.16

[BHHM20] BACK, JONGHEE, HUA, BINH-SON, HACHISUKA, TOSHIYA, and MOON, BOCHANG. "Deep Combiner for Independent and Correlated Pixel Estimates". *ACM Trans. Graph.* 39.6 (Nov. 2020). ISSN: 0730-0301. DOI: 10.1145/3414685.3417847. URL: https://doi.org/10.1145/3414685.3417847 1, 2, 5, 8–11.

[BHHM22] BACK, JONGHEE, HUA, BINH-SON, HACHISUKA, TOSHIYA, and MOON, BOCHANG. "Self-Supervised Post-Correction for Monte Carlo Denoising". *ACM SIGGRAPH 2022 Conference Proceedings*. SIGGRAPH '22. Vancouver, BC, Canada: Association for Computing Machinery, 2022. ISBN: 9781450393379. DOI: 10.1145/3528233.3530730. URL: https://doi.org/10.1145/3528233.3530730 8.

[BHHM23] BACK, JONGHEE, HUA, BINH-SON, HACHISUKA, TOSHIYA, and MOON, BOCHANG. "Input-Dependent Uncorrelated Weighting for Monte Carlo Denoising". *SIGGRAPH Asia 2023 Conference Papers*. SA '23. <conf-loc>, <city>Sydney</city>, <state>NSW</state>, <country>Australia</country>, </conf-loc>: Association for Computing Machinery, 2023. ISBN: 9798400703157. DOI: 10.1145/3610548.3618177. URL: https://doi.org/10.1145/3610548.3618177 8, 10.

[Bit16] BITTERLI, BENEDIKT. *Rendering resources*. https://benedikt-bitterli.me/resources/. 2016 5, 6, 10.

[BRM*16] BITTERLI, BENEDIKT, ROUSSELLE, FABRICE, MOON, BOCHANG, et al. "Nonlinearly weighted first-order regression for denoising Monte Carlo renderings". *Computer Graphics Forum*. Vol. 35. 4. Wiley Online Library. 2016, 107–117 3.

[BSH02] BEKAERT, PHILIPPE, SBERT, MATEU, and HALTON, JOHN H. "Accelerating Path Tracing by Re-Using Paths." *Rendering Techniques* 2.2002 (2002), 125–134 9.

[BVM*17a] BAKO, STEVE, VOGELS, THIJS, MCWILLIAMS, BRIAN, et al. "Kernel-predicting convolutional networks for denoising Monte Carlo renderings." *ACM Trans. Graph.* 36.4 (2017), 97–1 7.

[BVM*17b] BAKO, STEVE, VOGELS, THIJS, MCWILLIAMS, BRIAN, et al. "Kernel-predicting convolutional networks for denoising Monte Carlo renderings". *ACM Trans. Graph.* 36.4 (July 2017). ISSN: 0730-0301. DOI: 10.1145/3072959.3073708. URL: https://doi.org/10.1145/3072959.3073708 3.

[BWM*23] BALINT, MARTIN, WOLSKI, KRZYSZTOF, MYSZKOWSKI, KAROL, et al. "Neural Partitioning Pyramids for Denoising Monte Carlo Renderings". *ACM SIGGRAPH 2023 Conference Proceedings*. SIGGRAPH '23. <conf-loc>, <city>Los Angeles</city>, <state>CA</state>, <country>USA</country>, </conf-loc>: Association for Computing Machinery, 2023. ISBN: 9798400701597. DOI: 10.1145/3588432.3591562. URL: https://doi.org/10.1145/3588432.3591562 2, 3, 5, 8, 10, 12.

[BWP*20] BITTERLI, BENEDIKT, WYMAN, CHRIS, PHARR, MATT, et al. "Spatiotemporal reservoir resampling for real-time ray tracing with dynamic direct lighting". *ACM Transactions on Graphics (Proceedings of SIGGRAPH)* 39.4 (July 2020). DOI: 10/gg8xc7 3, 9.

[Che24] CHEN, CHUHAO. *mitsuba*. https://github.com/CzzzzH/mitsuba/tree/79de54dfddc831fda1f54dc7904b3449baf33454. 2024 5.

[CHH*24] CHOI, HAJIN, HONG, SEOKPYO, HA, INWOO, et al. "Online Neural Denoising with Cross-Regression for Interactive Rendering". *ACM Trans. Graph.* 43.6 (Nov. 2024). ISSN: 0730-0301. DOI: 10.1145/3687938. URL: https://doi.org/10.1145/3687938 8.

[CHL24] CHEN, CHUHAO, HE, YUZE, and LI, TZU-MAO. "Temporally Stable Metropolis Light Transport Denoising using Recurrent Transformer Blocks". *ACM Transactions on Graphics (Proceedings of SIGGRAPH)* 4 (2024) 2, 3, 5, 7, 8, 10, 11.

[CKS*17] CHAITANYA, CHAKRAVARTY R ALLA, KAPLANYAN, ANTON S, SCHIED, CHRISTOPH, et al. "Interactive reconstruction of Monte Carlo image sequences using a recurrent denoising autoencoder". *ACM Transactions on Graphics (TOG)* 36.4 (2017), 1–12 7.

[CWX*22] CHAN, KELVIN CK, WANG, XINTAO, XU, XIANGYU, et al. "BasicVSR++: Improving Video Super-Resolution with Enhanced Propagation and Alignment". *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2022 4.

[CWY*21] CHAN, KELVIN CK, WANG, XINTAO, YU, KE, et al. "BasicVSR: The Search for Essential Components in Video Super-Resolution and Beyond". *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2021, 4947–4956 4.

[DHC*21] DENG, XI, HAŠAN, MILOŠ, CARR, NATHAN, et al. "Path graphs: iterative path space filtering". *ACM Transactions on Graphics (TOG)* 40.6 (2021), 1–15 9.

[DMB*14] DELBRACIO, MAURICIO, MUSÉ, PABLO, BUADES, ANTONI, et al. "Boosting monte carlo rendering by ray histogram fusion". *ACM Trans. Graph.* 33.1 (Feb. 2014). ISSN: 0730-0301. DOI: 10.1145/2532708. URL: https://doi.org/10.1145/2532708 8.

[FFJ22] FIRMINO, ARTHUR, FRISVAD, JEPPE REVALL, and JENSEN, HENRIK WANN. "Progressive denoising of Monte Carlo rendered images". *Computer Graphics Forum*. Vol. 41. 2. Wiley Online Library. 2022, 1–11 8.

[FFJ23] FIRMINO, ARTHUR, FRISVAD, JEPPE REVALL, and JENSEN, HENRIK WANN. "Denoising-aware adaptive sampling for Monte Carlo ray tracing". *ACM SIGGRAPH 2023 Conference Proceedings*. 2023, 1–11 8.

[FRRW24] FIRMINO, ARTHUR, RAMAMOORTHI, RAVI, REVALL FRISVAD, JEPPE, and WANN JENSEN, HENRIK. "Practical error estimation for denoised monte carlo image synthesis". *ACM SIGGRAPH 2024 Conference Papers*. 2024, 1–10 8.

[GIM22] GU, JEONGMIN, IGLESIAS-GUITIAN, JOSE A., and MOON, BOCHANG. "Neural James-Stein Combiner for Unbiased and Biased Renderings". *ACM Trans. Graph.* 41.6 (Nov. 2022). ISSN: 0730-0301. DOI: 10.1145/3550454.3555496. URL: https://doi.org/10.1145/3550454.3555496 8.

[HKL*25] HEDSTROM, TREVOR, KETTUNEN, MARKUS, LIN, DAQI, et al. "ReSTIR BDPT: Bidirectional ReSTIR Path Tracing with Caustics". *ACM Trans. Graph.* 44.5 (Sept. 2025). ISSN: 0730-0301. DOI: 10.1145/3744898. URL: https://doi.org/10.1145/3744898 9.

[HSU19] HARIS, MUHAMMAD, SHAKHNAROVICH, GREGORY, and UKITA, NORIMICHI. "Recurrent Back-Projection Network for Video Super-Resolution". *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2019, 3897–3906 4.

[HSW*21] HU, EDWARD J., SHEN, YELONG, WALLIS, PHILLIP, et al. "LoRA: Low-Rank Adaptation of Large Language Models". *CoRR* abs/2106.09685 (2021). arXiv: 2106.09685. URL: https://arxiv.org/abs/2106.09685 6.

[IMF*21] IŞIK, MUSTAFA, MULLIA, KRISHNA, FISHER, MATTHEW, et al. "Interactive Monte Carlo denoising using affinity of neural features". *ACM Transactions on Graphics (TOG)* 40.4 (2021), 1–13 3, 8.

[Jak10] JAKOB, WENZEL. *Mitsuba renderer*. http://www.mitsuba-renderer.org. 2010 5, 6.

[KBS15] KALANTARI, NIMA KHADEMI, BAKO, STEVE, and SEN, PRADEEP. "A machine learning approach for filtering Monte Carlo noise." *ACM Trans. Graph.* 34.4 (2015), 122–1 7.

[KDB16] KELLER, ALEXANDER, DAHM, KEN, and BINDER, NIKOLAUS. "Path space filtering". *Monte Carlo and Quasi-Monte Carlo Methods: MCQMC, Leuven, Belgium, April 2014*. Springer. 2016, 423–436 9.

[KMA*15] KETTUNEN, MARKUS, MANZI, MARCO, AITTALA, MIIKA, et al. "Gradient-domain path tracing". *ACM Trans. Graph.* 34.4 (July 2015). ISSN: 0730-0301. DOI: 10.1145/2766997. URL: https://doi.org/10.1145/2766997 3, 9.

[KT25] KALOJANOV, JAVOR and THURSTON, KIMBALL. *Robust Average Networks for Monte Carlo Denoising*. 2025. arXiv: 2310.04080 [cs.GR]. URL: https://arxiv.org/abs/2310.04080 4.

[LKB*22] LIN*, DAQI, KETTUNEN*, MARKUS, BITTERLI, BENEDIKT, et al. "Generalized Resampled Importance Sampling: Foundations of ReSTIR". *ACM Transactions on Graphics (Proceedings of SIGGRAPH 2022)* 41.4 (July 2022). (*Joint First Authors), 75:1–75:23. ISSN: 0730-0301. DOI: 10.1145/3528223.3530158. URL: https://doi.org/10.1145/3528223.3530158 3, 9.

[LKL*13] LEHTINEN, JAAKKO, KARRAS, TERO, LAINE, SAMULI, et al. "Gradient-Domain Metropolis Light Transport". *ACM Trans. Graph.* 32.4 (2013) 9.

[Lum17] LUMBERYARD, AMAZON. *Amazon Lumberyard Bistro, Open Research Content Archive (ORCA)*. http://developer.nvidia.com/orca/amazon-lumberyard-bistro. July 2017. URL: http://developer.nvidia.com/orca/amazon-lumberyard-bistro 6.

[MDC*21] MANTIUK, RAFAŁ K., DENES, GYORGY, CHAPIRO, ALEXANDRE, et al. "FovVideoVDP: a visible difference predictor for wide field-of-view video". *ACM Trans. Graph.* 40.4 (July 2021). ISSN: 0730-0301. DOI: 10.1145/3450626.3459831. URL: https://doi.org/10.1145/3450626.3459831 6.

[MH20] MUNKBERG, JACOB and HASSELGREN, JON. "Neural denoising with layer embeddings". *Computer Graphics Forum*. Vol. 39. 4. Wiley Online Library. 2020, 1–12 2, 6, 7.

[MKD*16] MANZI, MARCO, KETTUNEN, MARKUS, DURAND, FRÉDO, et al. "Temporal gradient-domain path tracing". *ACM Transactions on Graphics (TOG)* 35.6 (2016), 1–9 9.

[MRKN20] MÜLLER, THOMAS, ROUSSELLE, FABRICE, KELLER, ALEXANDER, and NOVÁK, JAN. "Neural Control Variates". *ACM Trans. Graph.* 39.6 (Nov. 2020), 243:1–243:19. ISSN: 0730-0301. DOI: 10.1145/3414685.3417804. URL: https://doi.org/10.1145/3414685.3417804 2.

[MZV*20] MENG, XIAOXU, ZHENG, QUAN, VARSHNEY, AMITABH, et al. "Real-time Monte Carlo Denoising with the Neural Bilateral Grid". *Eurographics Symposium on Rendering - DL-only Track*. Ed. by DACHSBACHER, CARSTEN and PHARR, MATT. The Eurographics Association, 2020. ISBN: 978-3-03868-117-5. DOI: 10.2312/sr.20201133 7.

[NRN*23] NICOLET, BAPTISTE, ROUSSELLE, FABRICE, NOVÁK, JAN, et al. "Recursive Control Variates for Inverse Rendering". *Transactions on Graphics (Proceedings of SIGGRAPH)* 42.4 (Aug. 2023). DOI: 10.1145/3592139 2.

[OLK*21] OUYANG, Y., LIU, S., KETTUNEN, M., et al. "ReSTIR GI: Path Resampling for Real-Time Path Tracing". *Computer Graphics Forum* 40.8 (2021), 17–29. DOI: https://doi.org/10.1111/cgf.14378. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1111/cgf.14378. URL: https://onlinelibrary.wiley.com/doi/abs/10.1111/cgf.14378 9.

[RFB15] RONNEBERGER, OLAF, FISCHER, PHILIPP, and BROX, THOMAS. *U-Net: Convolutional Networks for Biomedical Image Segmentation*. 2015. arXiv: 1505.04597 [cs.CV]. URL: https://arxiv.org/abs/1505.04597 5.

[RJN16] ROUSSELLE, FABRICE, JAROSZ, WOJCIECH, and NOVÁK, JAN. "Image-space Control Variates for Rendering". *ACM Transactions on Graphics (Proceedings of SIGGRAPH Asia)* 35.6 (Dec. 2016), 169:1–169:12. DOI: 10/f9cphw 2, 3.

[RLG*20] RIOUX-LAVOIE, DAMIEN, LITALIEN, JOEY, GRUSON, ADRIEN, et al. "Delayed Rejection Metropolis Light Transport". *ACM Transactions on Graphics* 39.3 (Apr. 2020). DOI: 10.1145/3388538 5.

[RSÖ*22] RIOUX-LAVOIE, DAMIEN, SUGIMOTO, RYUSUKE, ÖZDEMIR, TÜMAY, et al. "A Monte Carlo Method for Fluid Simulation". *ACM Transactions on Graphics* 41.6 (Dec. 2022). DOI: 10.1145/3550454.3555450 2.

[SC20] SAWHNEY, ROHAN and CRANE, KEENAN. "Monte Carlo Geometry Processing: A Grid-Free Approach to PDE-Based Methods on Volumetric Domains". *ACM Trans. Graph.* 39.4 (2020) 2.

[SGH*22] SALAÜN, CORENTIN, GRUSON, ADRIEN, HUA, BINH-SON, et al. "Regression-based Monte Carlo integration". *ACM Trans. Graph.* 41.4 (July 2022). ISSN: 0730-0301. DOI: 10.1145/3528223.3530095. URL: https://doi.org/10.1145/3528223.3530095 2.

[SKW*17] SCHIED, CHRISTOPH, KAPLANYAN, ANTON, WYMAN, CHRIS, et al. "Spatiotemporal variance-guided filtering: real-time reconstruction for path-traced global illumination". *Proceedings of High Performance Graphics*. 2017, 1–12 7.

[SLK*24] SAWHNEY, ROHAN, LIN, DAQI, KETTUNEN, MARKUS, et al. "Decorrelating ReSTIR Samplers via MCMC Mutations". *ACM Transactions on Graphics (ToG)* (Jan. 2024), to appear. DOI: 10.1145/3629166. eprint: 2211.00166 9.

[VFBD17] VAN DE WOESTIJNE, JORAN, FREDERICKX, ROALD, BILLEN, NIELS, and DUTRÉ, PHILIP. "Temporal Coherence for Metropolis Light Transport." *EGSR (EI&I)*. 2017, 55–63 9.

[VRM*18] VOGELS, THIJS, ROUSSELLE, FABRICE, MCWILLIAMS, BRIAN, et al. "Denoising with kernel prediction and asymmetric loss functions". *ACM Transactions on Graphics (TOG)* 37.4 (2018), 1–15 2, 4, 5, 8.

[WCY*19] WANG, XINTAO, CHAN, KELVIN CK, YU, KE, et al. "EDVR: Video Restoration with Enhanced Deformable Convolutional Networks". *The IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*. 2019 4.

[WGGH20] WEST, REX, GEORGIEV, ILIYAN, GRUSON, ADRIEN, and HACHISUKA, TOSHIYA. "Continuous Multiple Importance Sampling". *ACM Transactions on Graphics (Proceedings of SIGGRAPH)* 39.4 (July 2020). DOI: 10.1145/3386569.3392436 9.

[WGH22] WEST, REX, GEORGIEV, ILIYAN, and HACHISUKA, TOSHIYA. "Marginal Multiple Importance Sampling". *SIGGRAPH Asia 2022 Conference Papers*. SA '22 Conference Papers. Daegu, Republic of Korea: Association for Computing Machinery, 2022. ISBN: 9781450394703. DOI: 10.1145/3550469.3555388. URL: https://doi.org/10.1145/3550469.3555388 9.

[WQH*24] WANG, QI, QIAO, PENGJU, HUO, YUCHI, et al. "Neural Kernel Regression for Consistent Monte Carlo Denoising". *ACM Trans. Graph.* 43.6 (Nov. 2024). ISSN: 0730-0301. DOI: 10.1145/3687949. URL: https://doi.org/10.1145/3687949 8.

[WYZ*25] WU, ZHIZHEN, YUAN, ZHILONG, ZUO, CHENYU, et al. "MoFlow: Motion-Guided Flows for Recurrent Rendered Frame Prediction". *ACM Trans. Graph.* 44.2 (Apr. 2025). ISSN: 0730-0301. DOI: 10.1145/3730400. URL: https://doi.org/10.1145/3730400 4.

[WZH*23] WU, ZHIZHEN, ZUO, CHENYU, HUO, YUCHI, et al. "Adaptive Recurrent Frame Prediction with Learnable Motion Vectors". *SIGGRAPH Asia 2023 Conference Papers*. SA '23. Sydney, NSW, Australia: Association for Computing Machinery, 2023. ISBN: 9798400703157. DOI: 10.1145/3610548.3618211. URL: https://doi.org/10.1145/3610548.3618211 4.

[XLG*24] XU, BING, LI, TZU-MAO, GEORGIEV, ILIYAN, et al. "Residual path integrals for re-rendering". *Computer Graphics Forum (Proceedings of EGSR)* 43.4 (2024) 3.

[XZW*19] XU, BING, ZHANG, JUNFEI, WANG, RUI, et al. "Adversarial Monte Carlo Denoising with Conditioned Auxiliary Feature". *ACM Transactions on Graphics (Proceedings of ACM SIGGRAPH Asia 2019)* 38.6 (2019), 224:1–224:12 7.

[YNL*21] YU, JIAQI, NIE, YONGWEI, LONG, CHENGJIANG, et al. "Monte Carlo denoising via auxiliary feature guided self-attention." *ACM Trans. Graph.* 40.6 (2021), 273–1 7.

[ZGZZ20] ZHONG, ZHIHANG, GAO, YE, ZHENG, YINQIANG, and ZHENG, BO. "Efficient spatio-temporal recurrent neural network for video deblurring". *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part VI 16*. Springer. 2020, 191–207 4.

[ZJL*15] ZWICKER, MATTHIAS, JAROSZ, WOJCIECH, LEHTINEN, JAAKKO, et al. "Recent advances in adaptive sampling and reconstruction for Monte Carlo rendering". *Computer graphics forum*. Vol. 34. 2. Wiley Online Library. 2015, 667–681 7.

[ZLK*24] ZHANG, SONG, LIN, DAQI, KETTUNEN, MARKUS, et al. "Area ReSTIR: Resampling for Real-Time Defocus and Antialiasing". *ACM Trans. Graph.* 43.4 (July 2024). ISSN: 0730-0301. DOI: 10.1145/3658210. URL: https://doi.org/10.1145/3658210 9.

[ZLY*21] ZENG, ZHENG, LIU, SHIQIU, YANG, JINGLEI, et al. "Temporally Reliable Motion Vectors for Real-time Ray Tracing". *Computer Graphics Forum*. Vol. 40. 2. Wiley Online Library. 2021, 79–90 4.

[ZRJ*15] ZIMMER, HENNING, ROUSSELLE, FABRICE, JAKOB, WENZEL, et al. "Path-space Motion Estimation and Decomposition for Robust Animation Filtering". *Computer Graphics Forum (Proceedings of EGSR)* 34.4 (June 2015). DOI: 10/f7mb34 4.