DOI: 10.1111/cgf.14406
Pacific Graphics 2021
E. Eisemann, K. Singh, and F.-L Zhang
(Guest Editors)

Volume 40 (2021), Number 7

Consistent Post-Reconstruction for Progressive Photon Mapping

Hajin Choi and Bochang Moon

Gwangju Institute of Science and Technology, South Korea

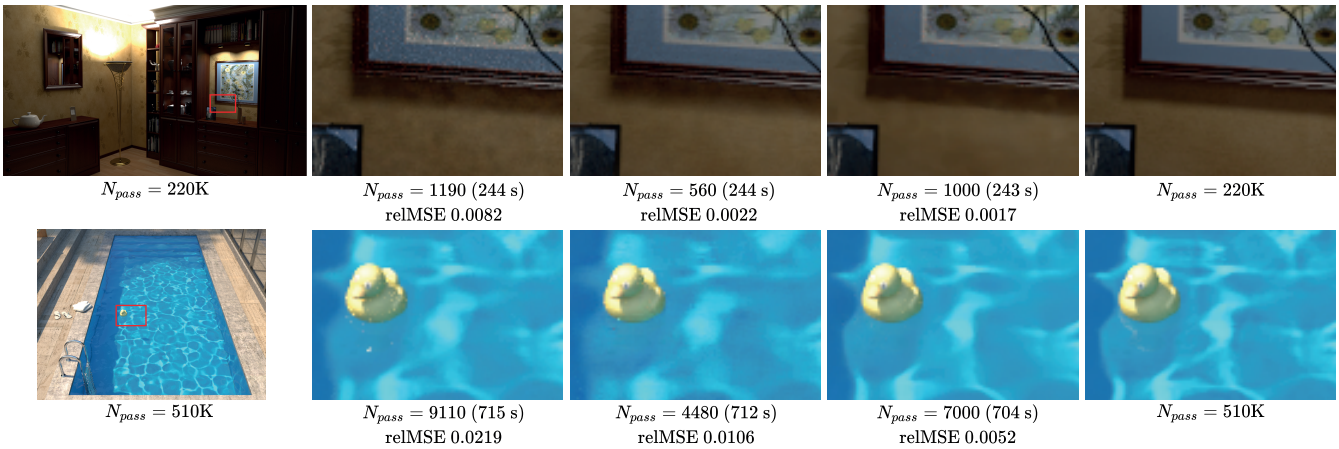


Figure 1: *Comparisons between two post-reconstruction techniques, single-buffered deep combiner (DC) [BHHM20] (c) and ours (d), which are integrated into stochastic progressive photon mapping (SPPM) [HJ09] (b). Both post-reconstruction techniques ((c) and (d)) effectively reduce the high-frequency noise in SPPM estimates, but our method produces sharper results than DC for the caustics (the bottom row). The number of iterations N_{pass}, where we use $0.1M$ photons per iteration, is adjusted so that each method uses approximately equal-render times, and we use relative mean-squared error (relMSE) [RKZ11] as a numerical measure.*

Abstract

Photon mapping is a light transport algorithm that simulates various rendering effects (e.g., caustics) robustly, and its progressive variants, progressive photon mapping (PPM) methods, can produce a biased but consistent rendering output. PPM estimates radiance using a kernel density estimation whose parameters (bandwidths) are adjusted progressively, and this refinement enables to reduce its estimation bias. Nonetheless, many iterations (and thus a large number of photons) are often required until PPM produces nearly converged estimates. This paper proposes a post-reconstruction that improves the performance of PPM by reducing residual errors in PPM estimates. Our key idea is to take multiple PPM estimates with multi-level correlation structures, and fuse the input images using a weight function trained by supervised learning with maintaining the consistency of PPM. We demonstrate that our technique boosts an existing PPM technique for various rendering scenes.

CCS Concepts

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

1. Introduction

Photon mapping (PM) [Jen96] has been recognized as one of the most effective global illumination methods since it can robustly simulate complex rendering phenomena such as caustics. It performs radiance estimation at hit points (e.g., intersection points between rays and a scene) where photons nearby the points are averaged through a kernel function with a bandwidth parameter. While the estimates can converge to the ground truth values given an infinite number of photons, in practice, this consistency cannot be accomplished due to limited memory.

Progressive photon mapping (PPM) [HOJ08, HJ09, KZ11] adapted the ordinary PM into a consistent method that generates

the correct estimates given an infinite number of photons, but without the memory requirement (i.e., infinite memory). PPM stores only accumulated statistics (e.g., accumulated photon flux) instead of keeping individual photons, and thus the memory overhead required is bounded. It allows shrinking the bandwidth of its kernel function iteratively and leads to the main strength of the algorithm, consistency.

It has been demonstrated that this progressive refinement of the kernel bandwidth can be further optimized by selecting the parameter adaptively [KD13, LLZ*20]. Nevertheless, it requires a nontrivial time until PPM produces nearly converged radiance estimates. Otherwise, the rendered images can exhibit noise or over-blurred artifacts in the estimates.

To reduce remaining errors in PPM estimates, a post-reconstruction can be applied to the images. As an example, deep combiner [BHHM20] enhanced reconstructed images through a combination function that blends its inputs, independent and correlated estimates (e.g., path-traced and reconstructed images). However, as shown in Fig. 1, this technique can produce suboptimal results for PPM estimates. It needs to generate its independent input image using a separate light transport algorithm (e.g., path tracing), leading to a significant increase in its computational overhead.

This paper proposes a more effective post-reconstruction that takes the output images of a PPM technique as input and generates an improved result. While the deep combiner previously addressed such a post-reconstruction problem, our post-reconstruction is specialized for PPM techniques. Our main idea is to generate multiple estimates with different smoothing levels by a chosen PPM method without relying on a separate light transport, e.g., path tracing. It allows us to mix these estimates with various structures more effectively than the recent post-reconstruction, as shown in Fig. 1. We demonstrate that our technique can improve the reconstruction results of existing PPM techniques (e.g., [HJ09, LLZ*20]) for various scenarios while maintaining the consistency of PPM.

2. Related Work

In this section, we briefly discuss photon mapping, its progressive variants, and image reconstruction related to our post-reconstruction.

Photon mapping. Photon mapping (PM) [Jen96, Jen09] is a two-pass rendering algorithm that traces photons from light sources and then produces radiance estimates at hit points through a kernel density estimation. Various optimization techniques have been proposed to improve the radiance estimation. Examples are GPU acceleration techniques for real-time density estimation [ZHWG08, MLM13], an optimal bandwidth selection for the kernel density estimation [Sch03], and anisotropic filters for the photon density estimation [SSFO08]. In addition, Qin et al. [QSH*15] demonstrated that a reconstruction bias introduced by the kernel density estimation could be removed by replacing the density estimation with a path connection that directly links the eye and light subpaths. Other notable examples include photon relaxation techniques [SJ09, SJ13] and photon beams for participating media [JZJ08].

Recently, Zhou et al. [ZXJ*20] proposed a learning-based density estimation that takes photons as input and outputs high-quality radiance estimates. We also propose a learning-based method, but our technique is a post-reconstruction that takes the results (i.e., images) of photon mapping methods as input, unlike the reconstruction method. It allows our method to be compatible with progressive photon mapping, where new photons are added per iteration.

Progressive photon mapping. Progressive photon mapping (PPM) [HOJ08] is a biased but consistent photon mapping that can generate correct radiance estimates with an infinite number of photons. PPM was extended to a generalized one, stochastic progressive photon mapping (SPPM) [HJ09], which shares photon statistics at the hit points generated from a pixel. The asymptotic errors (bias and variance) of PPM were analyzed in [HJJ10, KZ11], and PPM was extended into gradient-domain variants [HGNH17, GHV*18, XSW*20], which exploit estimated image gradients additionally. In addition, adaptive bandwidth selections [KD13, LLZ*20] were explored to balance the bias and variance of photon density estimation.

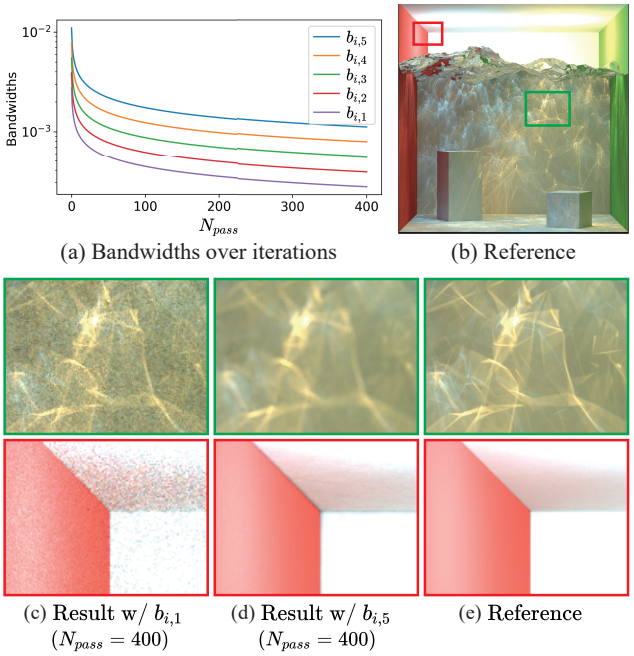
Nevertheless, these progressive methods can exhibit residual errors in their resulting estimates. Zeng et al. [ZWW*20] proposed a learning-based technique using multi-residual blocks, which alleviates such errors in PPM estimates. The objective of our method is also to reduce the residual errors in PPM estimates, but our technique maintains the consistency of the input estimates, unlike the recent work.

Reconstruction and post-reconstruction for Monte Carlo denoising. Removing noise in a rendered image has been actively studied, especially for Monte Carlo ray tracing [Kaj86]. We refer to a survey [ZJL*15] on this topic. Classical approaches (e.g., [MCY14, BRM*16]) using mean-squared error estimation and recent learning-based techniques (e.g., [BVM*17, VRM*18, XZW*19]) demonstrated effective noise reduction for path-traced images. One may apply these denoisers to PPM estimates for reducing its residual noise, but it cannot be effective since photon mapping estimates have both errors (bias and variance), unlike path-traced images.

One can consider a post-reconstruction technique, deep combiner (DC) [BHHM20], as an alternative to such image denoising. The recent method showed a performance improvement when it combines independent (e.g., path-traced images) and correlated estimates (e.g., their denoised images). However, adopting this approach for PPM estimates can introduce suboptimal results (e.g., in Fig. 1) since its independent input needs to be generated by an additional process, path tracing. We modify such a combination process into a more effective form that considers multi-level correlation structures in PPM estimates.

3. Background: Progressive Photon Mapping

This section provides a brief overview of progressive photon mapping (PPM) techniques. Our discussion focuses on stochastic progressive photon mapping (SPPM) [HJ09] that is an extended one of the ordinary PPM [HOJ08]. SPPM is an iterative algorithm that



(a) Bandwidths over iterations (b) Reference

(c) Result w/ $b_{i,1}$ ($N_{pass} = 400$) (d) Result w/ $b_{i,5}$ ($N_{pass} = 400$) (e) Reference

Figure 2: *SPPM estimates with different bandwidths. The plot (a) shows the bandwidths $b_{i,1},...,b_{i,5}$ (from the smallest to the largest) over iterations. We compute the bandwidths using a recursive rule (Eq. 4) but with different initial conditions, i.e., the bandwidths in the first pass. As an example, we set the initials to $1/4$, $1/2\sqrt{2}$, $1/2$, $1/\sqrt{2}$, 1 for the $b_{i,1},...,b_{i,5}$, respectively. Note that a change in the initial condition can lead to a noticeable difference in the estimates ((c) and (d)).*

updates its output estimates $\tilde{y}$ of the ground truth y using newly generated photons per iteration. Specifically, the estimate $\tilde{y}_c$ at pixel c can be written as a pixel estimation form [KZ11]:

$$\tilde{y}_c = \frac{1}{N_{pass}} \sum_{i=1}^{N_{pass}} \frac{f(x_i,\omega_i)}{p(x_i,\omega_i)} \hat{L}(x_i,\omega_i), \tag{1}$$

where $f(x_i,\omega_i)$ is the weight function (e.g., pixel reconstruction filters) that controls the relative contribution of the radiance estimate $\hat{L}(x_i,\omega_i)$ at the i-th hit point x_i. The $p(x_i,\omega_i)$ is the probabilistic density of the i-th eye subpath constructed by distributed ray tracing [CPC84].

The radiance estimate $\hat{L}(x_i,\omega_i)$ is computed using photons generated in the i-th pass, and it can be represented as a kernel density estimation [KZ11, KD13]:

$$\hat{L}(x_i,\omega_i) = \frac{1}{N_{photon}} \sum_{j=1}^{N_{photon}} K_{b_i}(x_j - x_i)\psi_j, \tag{2}$$

which averages the contribution of the j-th photon ψ_j, i.e., the photon value multiplied by the BRDF at x_i. This local averaging at x_i is controlled by a kernel function $K_{b_i}(\cdot)$ with a bandwidth parameter b_i, which adjusts the weight for the j-th photon at x_j. An example of the function is an isotropic kernel [HOJ08] that equally considers the photons whose Euclidean distances from the hit point x_i are less than the bandwidth b_i:

$$K_{b_i}(x_j - x_i) = \begin{cases} \frac{1}{\pi b_i^2} & \text{if } \|x_j - x_i\| < b_i \\ 0 & \text{otherwise.} \end{cases} \tag{3}$$

It is required to reduce the bandwidth b_i iteratively to make the estimation consistent, and SPPM uses a bandwidth update rule that can be written as a simple recursive form [KZ11]:

$$\frac{b_{i+1}^2}{b_i^2} = \frac{i+\alpha}{i+1}, \tag{4}$$

where α ($0 < \alpha < 1$) is a user-defined parameter that controls the reduction rate of the bandwidth. One can set the parameter to the asymptotically optimal one ($\alpha = 2/3$) [KD13]. The initial bandwidth b_1 in the recursion is typically determined using a k-nearest neighbor (k-NN) search with a user-defined parameter k.

The motivation of our post-reconstruction. One can further improve the photon mapping estimates $\hat{y}$ using an adaptive update rule that uses an estimated optimal bandwidth (e.g., [KD13, LLZ*20]). Nonetheless, the initial bandwidth b_1 has been determined heuristically, e.g., k-NN search with a user-defined k. Fig. 2 shows that SPPM estimates can be changed significantly by varying the initial bandwidth. In addition, PPM techniques often suffer from a high variance introduced by distributed ray tracing, especially for scenes that include glossy reflections. Technically, the ray-tracing noise cannot be effectively eliminated by the photon density estimation with a large bandwidth, since such noise is introduced by an independent process (i.e., distributed ray tracing). We aim to handle the technical challenges using a post-reconstruction (Sec. 4).

4. Post-Reconstruction for Progressive Photon Mapping

Our goal is to enhance the estimates of PPM techniques through a post-reconstruction that takes the estimates as input and produces an improved output. We generate multiple photon mapping estimates by varying the initial bandwidth b_1 instead of using a fixed one and then fuse the multi-level estimates via a combination guided by a neural network.

4.1. Generation of Multi-Level Estimates

Let us denote a series of the initial bandwidth b_1 as $b_{1,1},...,b_{1,m}$ (sorted in ascending order). To determine the largest one $b_{1,m}$, we exploit a k-NN search from the hit point x_i. Specifically, we search k nearest photons at the x_i and assign the Euclidean distance between the k-th photon and the x_i to the $b_{1,m}$. We set the k to a large one ($k = 20$). Once we set the largest one, the others are shrunk by a factor of $1/\sqrt{2}$. For example, the second and third largest ones ($b_{1,m-1}$ and $b_{1,m-2}$) become $b_{1,m}/\sqrt{2}$ and $b_{1,m}/2$. We use five bandwidths (i.e., $m = 5$).

We provide the initial bandwidths to an existing PPM technique so that it can generate multiple photon mapping estimates progressively using its bandwidth update rule (e.g., Eq. 4), which uses our initial bandwidths. Let us indicate that the sharpest estimates with the smallest one $b_{i,1}$, updated from its initial value $b_{1,1}$, as $\tilde{y}$. Also, the others are denoted as $\tilde{z}^1,...,\tilde{z}^{m-1}$, which correspond to

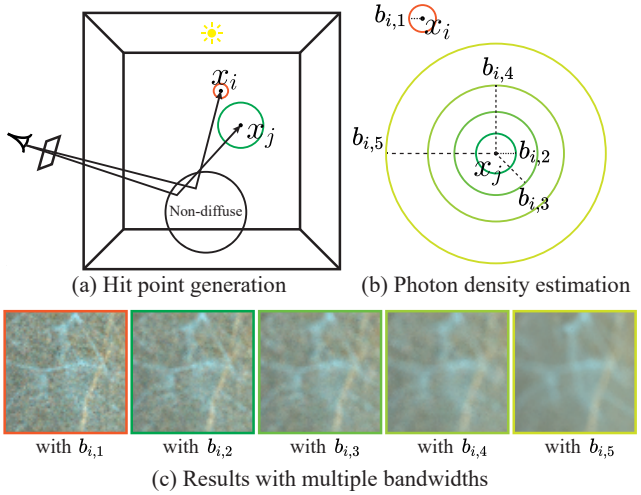


Figure 3: *Generation of multi-level estimates that are the input of our post-reconstruction. We generate two hit points x_i and x_j, using distributed ray tracing (a), and the points can be arbitrarily distant to each other when the eye subpaths contain a reflection on non-diffuse surfaces (e.g., glossy objects). At the first one x_i (in (b)), we generate the sharpest estimates $\tilde{y}$ using the smallest bandwidth $b_{i,1}$. On the other hand, the other estimates are generated using the second smallest to the largest one $b_{i,2}$,...,$b_{i,m}$ (e.g., $m = 5$). The resulting estimates (c) have different smoothing levels due to the bandwidths of various sizes.*

$b_{i,2}$,...,$b_{i,m}$, respectively. Intuitively, the $\tilde{y}$ is the nosiest but has the most negligible bias, and the $\tilde{z}^{m-1}$ has the slightest variance but highest bias. We treat the sharpest one as an approximately independent image and exploit the others as correlated images. Note that adjacent pixel colors in the photon mapping estimates are correlated when photons are shared for the radiance estimation, and the correlation in the sharpest estimates can be the lowest since the number of the shared photons often decreases with smaller bandwidths.

The straightforward implementation for generating the multi-level estimates is running a given PPM technique m times independently, but its computational overhead would increase linearly with the number of bandwidths. To mitigate the expensive overhead, we generate multiple estimates while sharing the photons (see Fig. 3). Note that we use the two different hit points x_i and x_j to decorrelate the independent estimates $\tilde{y}$ and the correlated estimates $\tilde{z}^1$,...,$\tilde{z}^{m-1}$. For example, the eye subpaths, which correspond to the hit points, can be divergent when a glossy reflection constructs the subpaths. It allows that the independent and correlated estimates have different ray tracing noise for the glossy reflection case.

Moreover, our multiple estimates have different bias and variance errors due to the bandwidths of various sizes. It enables our post-reconstruction to take advantage of the multi-level correlation structures in the estimates (in Sec. 4.2).

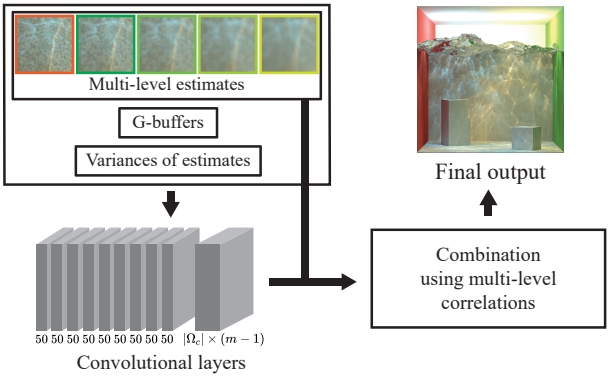


Figure 4: *Overview of our framework that takes multiple photon mapping estimates as input. We also feed the variances of the estimates and G-buffers (normals, albedos, and depth values) to the convolutional layers, and the last layer produces per-pixel weights for post-reconstruction. We fuse the input estimates with multi-level correlations using a combination function for producing the final output.*

Figure 5: *Example training images.*

4.2. Combination of Multiple Estimates

Once the input estimates $\tilde{y}$,$\tilde{z}^1$,...,$\tilde{z}^{m-1}$ are prepared, we combine the estimates to produce our final output $\hat{y}$. To this end, we adopt the multi-buffered combination [BHHM20] that combines the $m-1$ pairs of independent and correlated images:

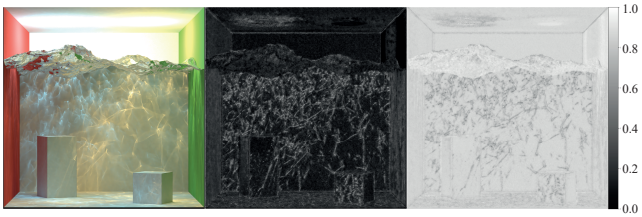
$$\frac{1}{W_c}\sum_{j=1}^{m-1}\left[\sum_{i\in\Omega_c} w_i^j \tilde{y}_i^j + \sum_{i\in\Omega_c} w_i^j\left(\tilde{z}_c^j - \tilde{z}_i^j\right)\right], \tag{5}$$

where $w_i^j > 0$ is the weight for $\tilde{y}_i^j$ and $\tilde{z}_i^j$ at pixel i in the j-th image. W_c is a normalization term, $W_c = \sum_{j=1}^{m-1}\sum_{i\in\Omega_c} w_i^j$, and Ω_c is a pixel set that includes all the pixels within a local window centered at pixel c. The combination function above exploits a positive correlation in correlated images through a difference term $(\tilde{z}_c^j - \tilde{z}_i^j)$.

Since we have only one independent image $\tilde{y}$ in our case, we assign it into the $m-1$ independent images $\tilde{y}^j$:

$$\hat{y}_c = \frac{1}{W_c}\sum_{j=1}^{m-1}\left[\sum_{i\in\Omega_c} w_i^j \tilde{y}_i + \sum_{i\in\Omega_c} w_i^j\left(\tilde{z}_c^j - \tilde{z}_i^j\right)\right]. \tag{6}$$

We employ the existing combination function, but the main differ-



(a) Our estimates $\hat{y}$ (b) $W_c^{-1}\sum_{i\in\Omega_c} w_i^1$ (c) $W_c^{-1}\sum_{i\in\Omega_c} w_i^4$

Figure 6: *Visualization of the relative importance of multiple SPPM estimates, where we show the sums of the combination weights for the sharpest (b) and smoothest estimates (c). Our method allocates more weights to the sharpest for high-frequency areas and gives higher weights to the smoothest for smooth areas.*

ence is that we generate the input estimates that have multi-level correlation structures. For example, Back et al. [BHHM20] generated multiple path-traced results and photon mapping estimates independently, using different random seeds. As a result, its correlated images have a similar amount of smoothing since the prior technique does not modify the initial bandwidths. On the other hand, our correlated images contain different smoothing levels due to the change in the initial bandwidth. It allows the combination function to select the different correlation structures more effectively using the combination weights w_i^j (in Eq. 6), as shown in an example figure (Fig. 6).

Consistency of our post-reconstruction. Our final estimate $\hat{y}_c$ goes to the ground-truth value y_c, as the number of iterations, N_{pass}, goes to the infinite:

$$\lim_{N_{pass}\to\infty} \hat{y}_c = y_c, \tag{7}$$

when our post-reconstruction takes consistent estimates as input. Appx. A includes our proof. Note that we vary only the initial bandwidth used in an existing PPM technique, and such a modification does not break the consistency of input progressive methods since the consistency depends on its bandwidth update rule (not its initial value). We refer to [KD13] that discusses the conditions required for the consistency of PPM. Technically, the consistency of our technique is irrespective of the weights w_i^j (in Eq. 6), and thus it enables us to choose the weights freely without sacrificing the asymptotic behavior.

4.3. Supervised Learning for Optimal Post-Reconstruction

To evaluate the combination function (Eq. 6), its parameters, the combination weights w_i^j, should be determined. We follow the deep combiner [BHHM20] that estimates proper weights using the kernel-predicting network [BVM*17]. Fig. 4 illustrates our framework that takes multi-level photon mapping estimates $\tilde{y}, \tilde{z}^1 ..., \tilde{z}^{m-1}$ ($m = 5$ in our setting) as input, and produces a final output $\hat{y}$. We additionally provide the variances of the estimates and G-buffers (normals, albedos, and depth values) to the network.

Specifically, we average the color variances to reduce the number of input channels, and thus the total dimension of the network input is 27 (5×3 for input estimates, 5 for their variances, and 7 for G-buffers). We use ten convolutional layers, and each intermediate layer uses 50 convolution filters of size 5×5. The last one uses $(m-1) \times |\Omega_c|$ filters of size 5×5 so that the combination weights can be produced per pixel. We set the size of the post-reconstruction window Ω_c to 19×19. The total number of trainable parameters is approximately 2.4M, given the network configuration. To train the neural network, we use a relative L_1 loss:

$$\mathcal{L} = \frac{1}{3N_{pixel}} \sum_{c=1}^{N_{pixel}} \sum_{l=1}^{3} \frac{\left|\log(\hat{y}_{c,l}+1) - \log(y_{c,l}+1)\right|}{\bar{y}_c + 0.01}, \tag{8}$$

where N_{pixel} is the total number of pixels in input images. $\hat{y}_{c,l}$ and $y_{c,l}$ are the l-th color channels of $\hat{y}_c$ and y_c respectively, and $\bar{y}_c = \frac{1}{3}\sum_{l=1}^{3} \log(y_{c,l}+1)$. The loss function $\mathcal{L}$ uses tone-mapped values, $\log(\hat{y}_{c,l}+1)$ and $\log(y_{c,l}+1)$, for stable learning since the $\hat{y}$ and y are high dynamic range (HDR) images.

Training details. We have exploited twelve public scenes provided by [Bit16], [Jak10] and [LLZ*20], and generated 60 scenes in total by randomizing the camera and materials of the public scenes. Fig. 5 shows examples of the randomized scenes. Given the scenes, we have generated the network inputs (multiple estimates, their variances, and G-buffers) using SPPM [HJ09]. Precisely, the variances have been calculated by Welford's online algorithm [Wel62], and the SPPM has used 0.1M photons per iteration. We have selected three iterations ($N_{pass} = \{100, 200, 400\}$), and produced ten images per iteration by changing the random seed. As a result, we have exploited 1800 ($60 \times 3 \times 10$) training data. The training images have been split into smaller images of size 64×64 and used for our supervised learning. For the reference images, we have employed SPPM estimates generated with large numbers of iterations (e.g., $N_{pass} = 220$K). We have initialized the network parameters using Glorot uniform initializer [GB10] and then trained it for 100 epochs (approximately 16 hours on two NVIDIA RTX 3090 given our implementation with Tensorflow [AAB*15]) using Adam optimizer [KB15] with the initial learning rate of 0.0001.

5. Results and Discussion

We evaluate our post-reconstruction with the two PPM techniques, stochastic progressive photon mapping (SPPM) [HJ09] and chi-squared progressive photon mapping (CPPM) [LLZ*20]. Note that we have trained our neural network only with training images generated by SPPM (in Sec. 4.3), and thus our post-reconstruction for CPPM is to verify whether our technique can improve the performance of the unseen progressive method. Specifically, we generate multi-level estimates for the combined techniques (Ours + SPPM and Ours + CPPM) by varying the initial bandwidth used in SPPM and CPPM, as described in Sec. 4.1. We assign the $b_{1,2}$, which is roughly corresponding to the one computed by the k-nearest neighbor search with $k = 10$, to the initial bandwidth of the previous techniques (SPPM and CPPM) without our method. The tested implementations of the progressive methods (e.g., CPPM implementation provided by the authors) use the Mitsuba renderer [Jak10]. We have amended the public implementation to use direct lighting when primary rays intersect with glossy surfaces since we have observed that this simple extension reduces ray tracing noise on glossy objects for both SPPM and CPPM.

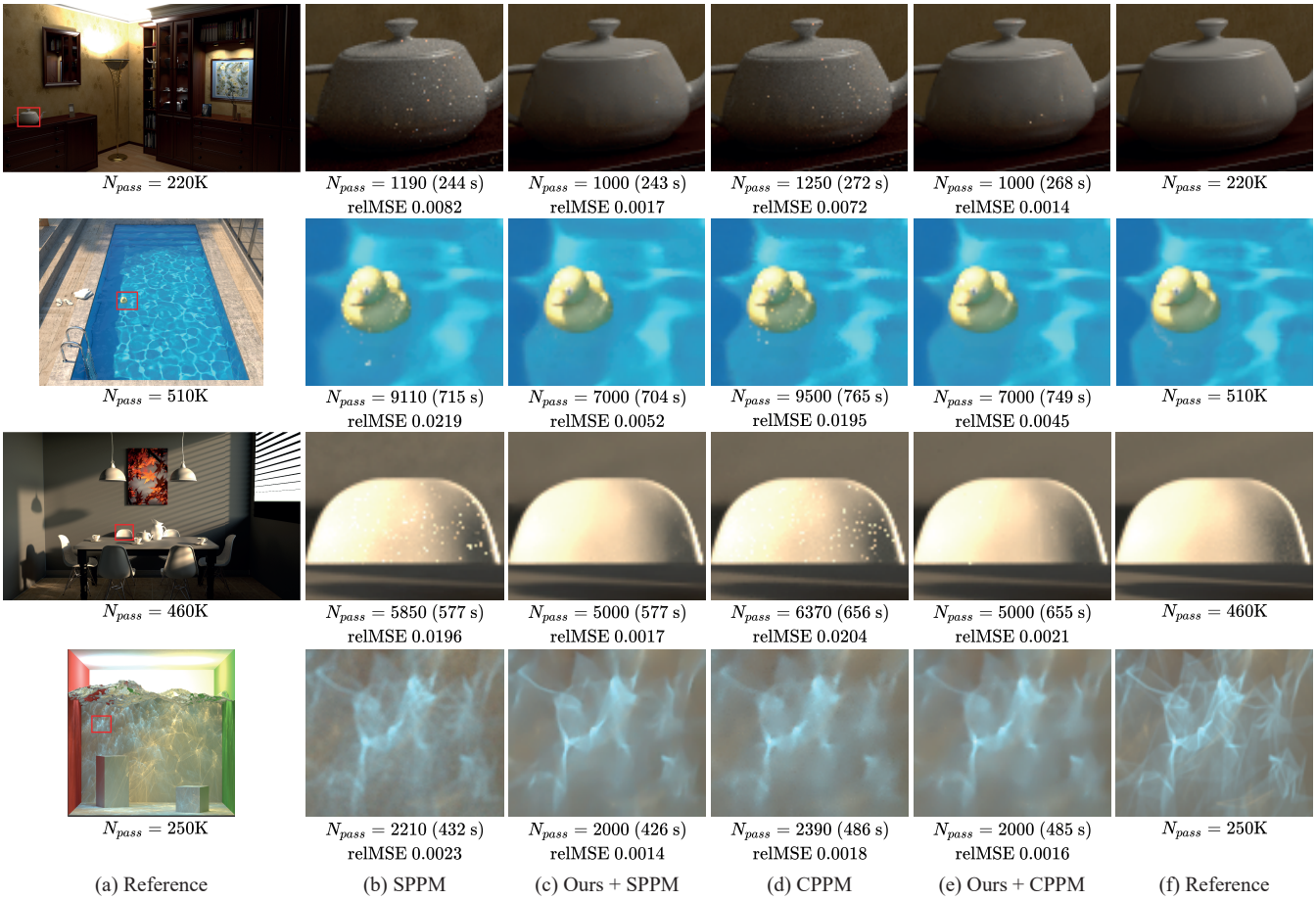


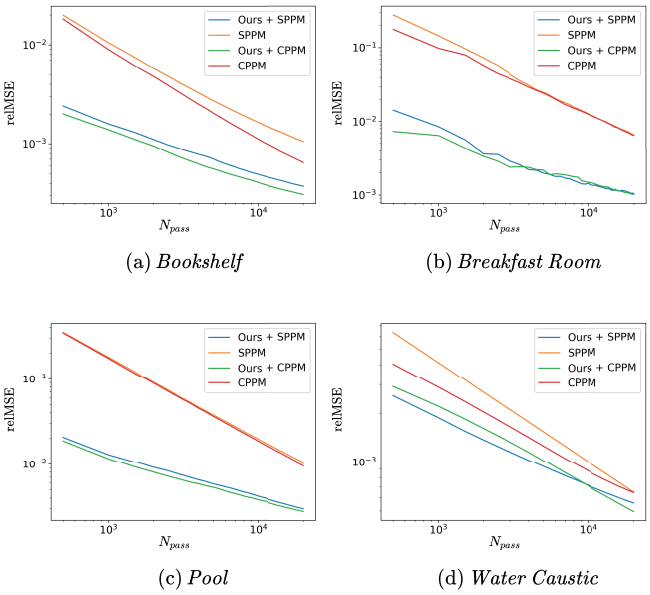
Figure 7: *Equal-time comparisons with SPPM and CPPM. We integrate our post-reconstruction into the progressive methods, so that improved estimates can be generated. The PPM techniques (SPPM (b) and CPPM (d)) exhibit high-frequency noise for the Bookshelf and Breakfast Room scenes (in the top and third rows) and produce over-blurred artifacts for the Pool and Water Caustic scenes (in the second and bottom rows). Our technique ((c) and (e)) mitigates such artifacts while improving their numerical accuracy.*

We also compare our post-reconstruction with deep combiner (DC). Precisely, we show its single-buffered or multi-buffered versions (referred to as DC (single) and DC (multi)), which use single-pair or four-pairs of independent and correlated images, respectively. To generate their independent images, we use bidirectional path tracing (BDPT) [LW93, VG95] and adjust the rendering time of BDPT to be approximately equal to generating its correlated images. For a fair comparison, we have retrained the DC (single) and DC (multi) using an extensive data set that includes their original and our training scenes, respectively.

The relative mean-squared error (relMSE) [RKZ11] is used for measuring the numerical accuracy of tested methods, and those errors are computed using reference images generated by SPPM with a large number of iterations. We use 0.1M photons per iteration. All tests are conducted using the Mitsuba [Jak10] renderer on a desktop with a CPU (AMD Ryzen 3990X) and GPU (NVIDIA RTX 3090).

We test the four scenes, *Bookshelf* (1280 × 720), *Pool* (800 × 600), *Breakfast Room* (1280 × 720), and *Water Caustic* (1024 × 1024), shown from top to bottom in Fig. 7. The numbers in the parentheses are the image resolutions for the scenes. In the *Bookshelf* and *Breakfast Room* scenes, most scene regions are lit by indirect illumination. Also, the *Pool* and *Water Caustic* scenes include strong caustics. The test scenes were not included in our training.

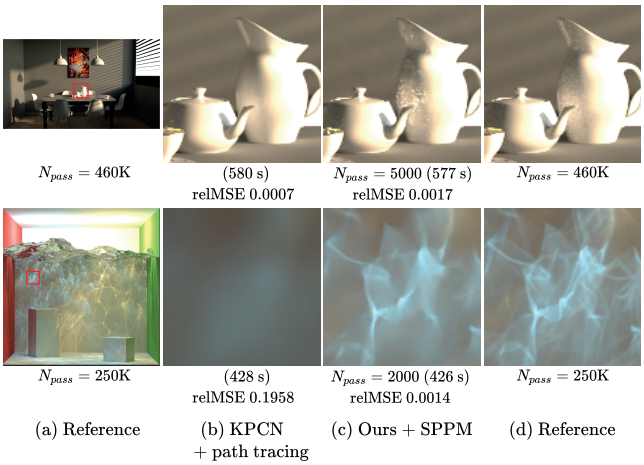
Comparisons with PPM techniques. Fig. 7 shows our post-reconstruction results with SPPM and CPPM. The progressive methods without our post-reconstruction show high-frequency noise on glossy surfaces (e.g., the top and third rows in the figure) since their radiance estimation cannot remove such noise introduced by distributed ray tracing. Our technique effectively reduces the noise for the methods and enhances their numerical accuracy. In addition, the results using our technique are sharper than those without ours for the caustics scenes. We also compare SPPM and CPPM with and without our post-correction by varying the number of photon iterations in Fig. 8. As shown in the figure, our technique enhances the numerical accuracy of the progressive meth-



(a) *Bookshelf* (b) *Breakfast Room*

(c) *Pool* (d) *Water Caustic*

Figure 8: *Numerical accuracy of SPPM and CPPM with and without our technique (shown in a log-log scale).*



(a) Reference (b) KPCN + path tracing (c) Ours + SPPM (d) Reference

Figure 9: *Equal-time comparisons with KPCN that denoises path-traced estimates. KPCN shows smoother results than our technique with SPPM for the Breakfast Room scene, but it does not preserve the strong caustics for the Water Caustic scene.*

	Bookshelf	*Pool*	*Breakfast R.*	*Water C.*
N_{pass}	1000	7000	5000	2000
Dual $(\bar{y}, \bar{z}^1)$	0.0026	0.0061	0.0018	0.0021
Dual $(\bar{y}, \bar{z}^2)$	0.0020	0.0061	0.0018	0.0018
Dual $(\bar{y}, \bar{z}^3)$	0.0017	0.0063	0.0018	0.0017
Dual $(\bar{y}, \bar{z}^4)$	0.0017	0.0066	0.0018	0.0018
Multi $(\bar{y}, \bar{z}^1, ..., \bar{z}^4)$	0.0017	0.0052	0.0017	0.0014

Table 1: *Numerical accuracy of our method with different numbers of input estimates, generated by SPPM with the iteration counts in Fig. 7. The tested variant of our method, Dual(·), exploits only two correlation levels, i.e., a single pair of independent and correlated estimates.*

ods over the tested range, thanks to the consistency of our post-reconstruction (in Sec. 4.2).

Comparisons with Monte Carlo denoising. In Fig. 9, we compare our method with an alternative that denoises path-traced estimates using a recent Monte Carlo denoiser, kernel-predicting convolutional networks (KPCN) [BVM*17]. We have fine-tuned its pre-trained neural network, provided by the respective authors, using our training dataset for a fair comparison. While the alternative shows a better output than our technique with SPPM for the *Breakfast Room* scene, it shows over-blurred artifacts on the caustics area for the *Water Caustic* scene. On the other hand, our method significantly improves the latter case since our input technique (SPPM) is more robust than path tracing for the caustics.

Comparisons with DC. Fig. 10 compares our technique with the previous post-reconstruction, DC, given a progressive method, SPPM. As can be seen in the figure, DC (single) and DC (multi) use smaller iterations than our method given the equal-time budgets since the prior should generate their independent images through an additional process (BDPT). DC (single and multi) effectively enhances the performance of SPPM, mainly when BDPT generates high-quality images for the independent inputs to the method (in the top and third rows of the figure). For example, DC (single) produces a slightly lower error than our method for the *Breakfast Room* scene.

Nonetheless, it fails to improve the caustic areas where BDPT does not produce enough information on the details (especially in the bottom row). On the other hand, we improve the visual quality and numerical accuracy of SPPM for the tested scenes, including the caustic scenes. Our robust behavior is mainly because that our technique generates multi-level estimates only using the progressive method while adjusting its smoothing parameters.

Ablation study. Our technique uses separate hit points x_i and x_j (in Fig. 3) where we generate independent $\bar{y}$ and correlated estimates $\bar{z}^1, ..., \bar{z}^{m-1}$, in order to decorrelate the ray tracing noise in the two types of input estimates. Fig. 11 compares our choice with an alternative that does not separate the eye subpaths. As shown in the figure, this separation enables us to remove the ray tracing noise effectively since our combination can down-weight either independent or correlated estimates.

We also test a variant of our method that uses only two input images, an independent $\bar{y}$ and only a correlated image (in Table 1). Specifically, we select its correlated image by picking only one from our correlated images in turn. As shown in the figure, the best level for the variant differs across the scenes, and it can be tricky to choose a proper one in practice. On the other hand, our choice of using multiple correlated estimates alleviates the difficulty.

Computational overhead. We report the runtime overheads of our post-reconstruction in Table 2. The most expensive part in our

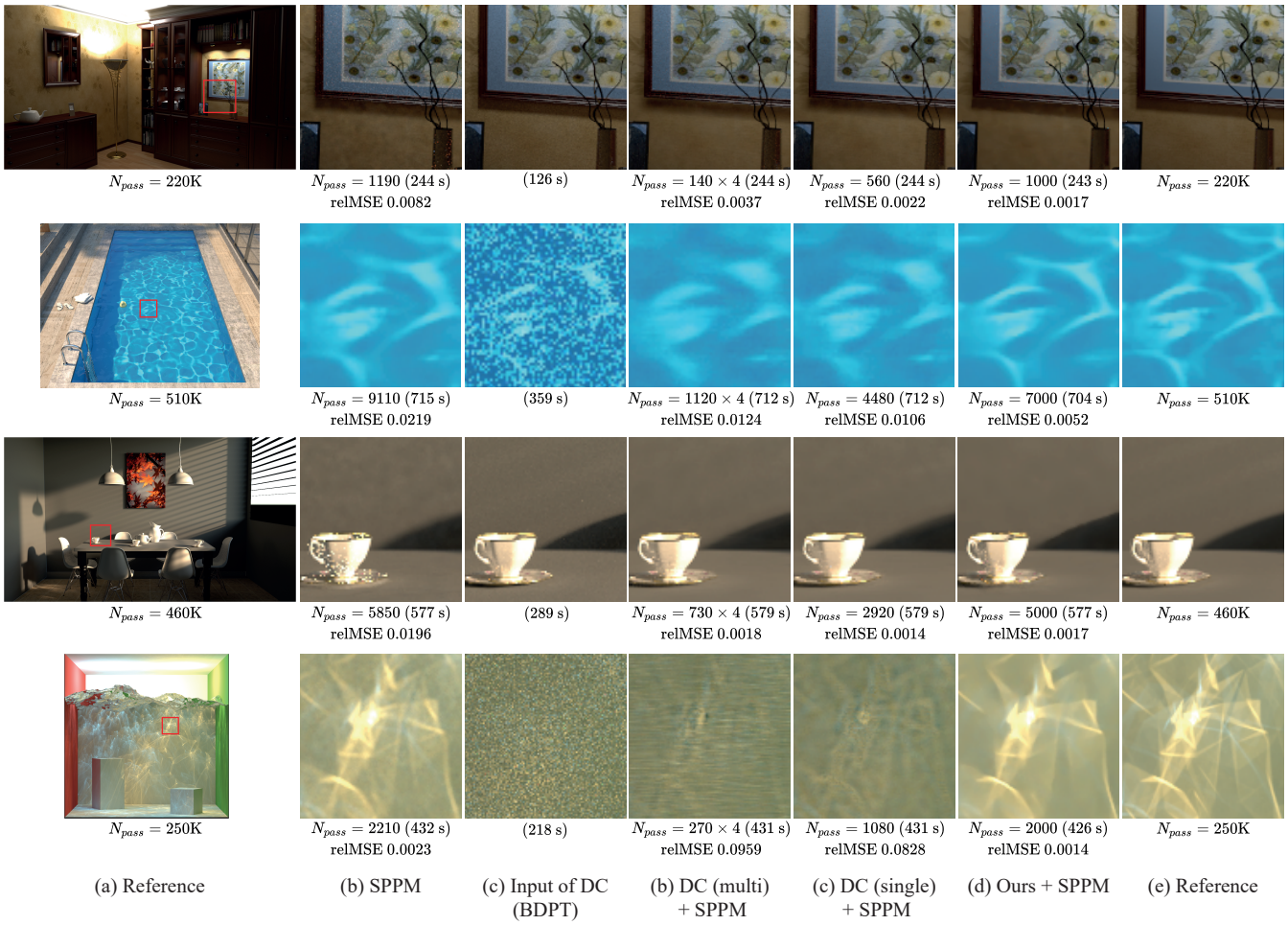


Figure 10: *Comparisons with DC. DC (single) uses an independent image (c), and DC (multi) uses four independent images generated by BDPT. The image (c) corresponds to the average of the four images for DC (multi). DC (multi) uses* $4\times$ *smaller iterations than DC (single) given the equal-time budget. The previous technique (DC (single) and DC (multi)) produces high-quality reconstruction results when BDPT gives high-quality independent input to the method (the top and third rows). Nevertheless, it smooths the caustics overly (the second and bottom rows) that are not captured effectively by BDPT. On the other hand, our method reduces the residual errors of SPPM consistently.*

post-reconstruction is generating multiple input estimates, but as shown in the table, it does not drastically increase the render times per iteration since the input images are generated while sharing photon maps. As a result, our overheads over the input methods (SPPM and CPPM) are in the moderate range of 8.67% to 32.10%.

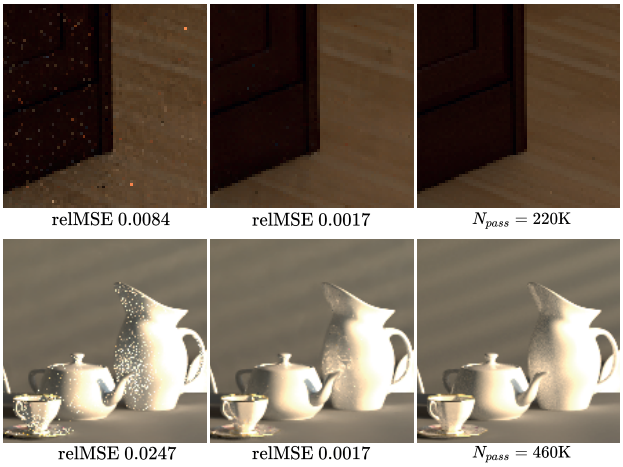
Limitations and future work. Our technique takes multiple estimates generated by a progressive method, and thus the quality of our final output relies on the input estimates. As shown in Fig. 12, our method can fail to preserve high-frequency details (e.g., caustics) when all the input estimates do not have the fine details appropriately. To mitigate such a problem, we would like to design a unified framework that optimizes the bandwidth update rule of a progressive technique while considering the errors in our final image. In addition, exploiting temporal coherence in animated sequences can be considered as future work. It is also interesting to test our post-reconstruction with gradient-domain variants (e.g., [HGNH17]) for enhanced output.

6. Conclusions

In this paper, we have presented a post-reconstruction technique that reduces remaining errors in PPM estimates without sacrificing the consistency of the input methods. We generate multiple input estimates by feeding different initial bandwidths to a progressive method, and it allows us to take into account multi-level correlation structures through a combination process. Besides, we produce independent and correlated images using two separate hit points to decorrelate ray tracing noise in the two input types.

Acknowledgements

We appreciate the anonymous reviewers for the constructive comments. We also thank the following authors and artists for each



(a) W/o decorrelation (b) W/ decorrelation (c) Reference

Figure 11: *Comparisons with an alternative that does not decorrelate the independent and correlated estimates, generated by SPPM with $N_{pass} = 1K$ (top) and $N_{pass} = 5K$ (bottom). The alternative (a) leaves the high-frequency noise since its input images share the noise generated by distributed ray tracing. Our decorrelation (b), however, allows us to reduce such noise adequately.*

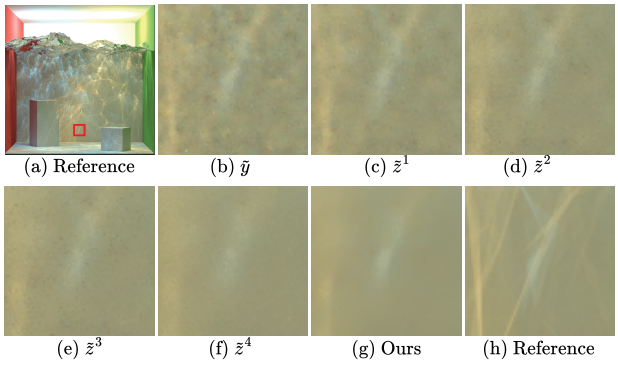


Figure 12: *Failure case of our post-reconstruction for CPPM with $N_{pass} = 400$. Our input estimates ((b) to (f)), generated by CPPM, do not fruitfully contain the high-frequency information (i.e., caustics) and lead to over-blurred artifacts in our final output.*

	Bookshelf	*Pool*	*Breakfast R.*	*Water C.*
		SPPM		
N_{pass}	1190	9110	5850	2210
Total	244.098 s	714.766 s	577.194 s	432.234 s
Total / N_{pass}	0.205 s	0.078 s	0.099 s	0.196 s
		Ours + SPPM		
N_{pass}	1000	7000	5000	2000
Input	241.840 s	703.828 s	576.707 s	425.118 s
Inference	0.733 s	0.302 s	0.733 s	0.839 s
Total	242.573 s	704.130 s	577.440 s	425.957 s
Total / N_{pass}	0.243 s	0.101 s	0.115 s	0.213 s
Overhead	18.54%	29.49%	16.16%	8.67%
		CPPM		
N_{pass}	1250	9500	6370	2390
Total	271.875 s	765.471 s	656.193 s	485.592 s
Total / N_{pass}	0.218 s	0.081 s	0.103 s	0.203 s
		Ours + CPPM		
N_{pass}	1000	7000	5000	2000
Input	267.006 s	748.581 s	653.820 s	483.849 s
Inference	0.733 s	0.302 s	0.733 s	0.839 s
Total	267.739 s	748.883 s	654.553 s	484.688 s
Total / N_{pass}	0.268 s	0.107 s	0.131 s	0.242 s
Overhead	22.94%	32.10%	27.18%	19.21%

Table 2: *Computational overheads of SPPM and CPPM with and without our technique. The statistics are measured from the equal-time comparisons (in Fig. 7).*

scene: *Bookshelf* (Tiziano Portenier), *Breakfast Room* (Wig42), *Pool* (Ondřej Karlík) and *Water Caustic* (generated by Benedikt Bitterli and modified by the authors of [LLZ*20]). Bochang Moon is the corresponding author of the paper. This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (No. 2020R1A2C4002425).

References

[AAB*15] ABADI M., AGARWAL A., BARHAM P., BREVDO E., CHEN Z., CITRO C., CORRADO G. S., DAVIS A., DEAN J., DEVIN M., GHEMAWAT S., GOODFELLOW I., HARP A., IRVING G., ISARD M., JIA Y., JOZEFOWICZ R., KAISER L., KUDLUR M., LEVENBERG J., MANÉ D., MONGA R., MOORE S., MURRAY D., OLAH C., SCHUSTER M., SHLENS J., STEINER B., SUTSKEVER I., TALWAR K., TUCKER P., VANHOUCKE V., VASUDEVAN V., VIÉGAS F., VINYALS O., WARDEN P., WATTENBERG M., WICKE M., YU Y., ZHENG X.: TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software available from tensorflow.org.

[BHHM20] BACK J., HUA B.-S., HACHISUKA T., MOON B.: Deep combiner for independent and correlated pixel estimates. *ACM Trans. Graph. 39*, 6 (Nov. 2020).

[Bit16] BITTERLI B.: Rendering resources, 2016.

[BRM*16] BITTERLI B., ROUSSELLE F., MOON B., IGLESIAS-GUITIÁN J. A., ADLER D., MITCHELL K., JAROSZ W., NOVÁK J.: Nonlinearly weighted first-order regression for denoising Monte Carlo renderings. *Computer Graphics Forum 35*, 4 (July 2016), 107–117.

[BVM*17] BAKO S., VOGELS T., MCWILLIAMS B., MEYER M., NOVÁK J., HARVILL A., SEN P., DEROSE T., ROUSSELLE F.: Kernel-predicting convolutional networks for denoising Monte Carlo renderings. *ACM Trans. Graph. 36*, 4 (July 2017).

[CPC84] COOK R. L., PORTER T., CARPENTER L.: Distributed ray tracing. *SIGGRAPH Comput. Graph. 18*, 3 (Jan. 1984), 137–145.

[GB10] GLOROT X., BENGIO Y.: Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics* (May 2010), pp. 249–256.

[GHV*18] GRUSON A., HUA B.-S., VIBERT N., NOWROUZEZAHRAI D., HACHISUKA T.: Gradient-domain volumetric photon density estimation. *ACM Trans. Graph. 37*, 4 (July 2018).

[HGNH17] HUA B.-S., GRUSON A., NOWROUZEZAHRAI D., HACHISUKA T.: Gradient-domain photon density estimation. *Computer Graphics Forum 36*, 2 (2017), 31–38.

[HJ09] HACHISUKA T., JENSEN H. W.: Stochastic progressive photon mapping. *ACM Trans. Graph. 28*, 5 (Dec. 2009), 1–8.

[HJJ10] HACHISUKA T., JAROSZ W., JENSEN H. W.: A progressive error estimation framework for photon density estimation. *ACM Trans. Graph. 29*, 6 (Dec. 2010).

[HOJ08] HACHISUKA T., OGAKI S., JENSEN H. W.: Progressive photon mapping. *ACM Trans. Graph. 27*, 5 (Dec. 2008).

[Jak10] JAKOB W.: Mitsuba renderer, 2010.

[Jen96] JENSEN H. W.: Global illumination using photon maps. In *Proceedings of the Eurographics Workshop on Rendering Techniques '96* (1996), Springer-Verlag, p. 21–30.

[Jen09] JENSEN H. W.: *Realistic Image Synthesis Using Photon Mapping*. A. K. Peters, Ltd., 2009.

[JZJ08] JAROSZ W., ZWICKER M., JENSEN H. W.: The beam radiance estimate for volumetric photon mapping. *Computer Graphics Forum 27*, 2 (Apr. 2008), 557–566.

[Kaj86] KAJIYA J. T.: The rendering equation. *SIGGRAPH Comput. Graph. 20*, 4 (Aug. 1986), 143–150.

[KB15] KINGMA D. P., BA J.: Adam: A method for stochastic optimization. In *ICLR (Poster)* (2015).

[KD13] KAPLANYAN A. S., DACHSBACHER C.: Adaptive progressive photon mapping. *ACM Trans. Graph. 32*, 2 (Apr. 2013).

[KZ11] KNAUS C., ZWICKER M.: Progressive photon mapping: A probabilistic approach. *ACM Trans. Graph. 30*, 3 (May 2011).

[LLZ*20] LIN Z., LI S., ZENG X., ZHANG C., JIA J., WANG G., MANOCHA D.: CPPM: Chi-squared progressive photon mapping. *ACM Trans. Graph. 39*, 6 (Nov. 2020).

[LW93] LAFORTUNE E. P., WILLEMS Y. D.: Bi-directional path tracing. In *Proceedings of Third International Conference on Computational Graphics and Visualization Techniques* (Dec. 1993), pp. 145–153.

[MCY14] MOON B., CARR N., YOON S.-E.: Adaptive rendering based on weighted local regression. *ACM Trans. Graph. 33*, 5 (Sept. 2014).

[MLM13] MARA M., LUEBKE D., MCGUIRE M.: Toward practical real-time photon mapping: Efficient gpu density estimation. In *Proceedings of the ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games* (2013), I3D '13, p. 71–78.

[QSH*15] QIN H., SUN X., HOU Q., GUO B., ZHOU K.: Unbiased photon gathering for light transport simulation. *ACM Trans. Graph. 34*, 6 (Oct. 2015).

[RKZ11] ROUSSELLE F., KNAUS C., ZWICKER M.: Adaptive sampling and reconstruction using greedy error minimization. *ACM Trans. Graph. 30*, 6 (Dec. 2011), 1–12.

[Sch03] SCHREGLE R.: Bias compensation for photon maps. *Computer Graphics Forum 22*, 4 (2003), 729–742.

[SJ09] SPENCER B., JONES M.: Into the blue: Better caustics through photon relaxation. *Computer Graphics Forum 28*, 2 (2009), 319–328.

[SJ13] SPENCER B., JONES M. W.: Progressive photon relaxation. *ACM Trans. Graph. 32*, 1 (Feb. 2013).

[SSFO08] SCHJØTH L., SPORRING J., FOGH OLSEN O.: Diffusion based photon mapping. *Computer Graphics Forum 27*, 8 (2008), 2114–2127.

[VG95] VEACH E., GUIBAS L.: Bidirectional estimators for light transport. In *Photorealistic Rendering Techniques*. Springer, 1995, pp. 145–167.

[VRM*18] VOGELS T., ROUSSELLE F., MCWILLIAMS B., RÖTHLIN G., HARVILL A., ADLER D., MEYER M., NOVÁK J.: Denoising with kernel prediction and asymmetric loss functions. *ACM Trans. Graph. 37*, 4 (July 2018).

[Wel62] WELFORD B. P.: Note on a method for calculating corrected sums of squares and products. *Technometrics 4*, 3 (1962), 419–420.

[XSW*20] XU Z., SUN Q., WANG L., XU Y., WANG B.: Unsupervised image reconstruction for gradient-domain volumetric rendering. *Computer Graphics Forum 39*, 7 (2020), 193–203.

[XZW*19] XU B., ZHANG J., WANG R., XU K., YANG Y.-L., LI C., TANG R.: Adversarial Monte Carlo denoising with conditioned auxiliary feature modulation. *ACM Trans. Graph. 38*, 6 (Nov. 2019).

[ZHWG08] ZHOU K., HOU Q., WANG R., GUO B.: Real-time kd-tree construction on graphics hardware. *ACM Trans. Graph. 27*, 5 (Dec. 2008).

[ZJL*15] ZWICKER M., JAROSZ W., LEHTINEN J., MOON B., RAMAMOORTHI R., ROUSSELLE F., SEN P., SOLER C., YOON S.-E.: Recent advances in adaptive sampling and reconstruction for Monte Carlo rendering. *Computer Graphics Forum 34*, 2 (2015), 667–681.

[ZWW*20] ZENG Z., WANG L., WANG B., KANG C., XU Y.: Denoising stochastic progressive photon mapping renderings using a multi-residual network. *Journal of Computer Science and Technology 35* (2020), 506–521.

[ZXJ*20] ZHU S., XU Z., JENSEN H. W., SU H., RAMAMOORTHI R.: Deep kernel density estimation for photon mapping. *Computer Graphics Forum 39*, 4 (2020), 35–45.

Appendix A: Consistency of our post-reconstruction

Recall that our combination (Eq. 6) is represented as

$$\hat{y}_c = \frac{1}{W_c}\sum_{j=1}^{m-1}\left[\sum_{i\in\Omega_c} w_i^j\left(\tilde{y}_i + \tilde{z}_c^j - \tilde{z}_i^j\right)\right]. \tag{9}$$

Let us consider a pixel $i = I_u$ and image index $j = J_u$ that make the $(\tilde{y}_i + \tilde{z}_c^j - \tilde{z}_i^j)$ the maximum, i.e., $(\tilde{y}_i + \tilde{z}_c^j - \tilde{z}_i^j) \le (\tilde{y}_{I_u} + \tilde{z}_c^{J_u} - \tilde{z}_{I_u}^{J_u})$ for all i and j. Then, an upper bound of the estimate $\hat{y}_c$ can be obtained as the following:

$$\begin{aligned}\hat{y}_c &\le \frac{1}{W_c}\sum_{j=1}^{m-1}\left[\sum_{i\in\Omega_c} w_i^j\left(\tilde{y}_{I_u} + \tilde{z}_c^{J_u} - \tilde{z}_{I_u}^{J_u}\right)\right] \\ &= \tilde{y}_{I_u} + \tilde{z}_c^{J_u} - \tilde{z}_{I_u}^{J_u}.\end{aligned} \tag{10}$$

The limit of the upper bound is as follows.

$$\lim_{N_{pass}\to\infty}\left(\tilde{y}_{I_u} + \tilde{z}_c^{J_u} - \tilde{z}_{I_u}^{J_u}\right) = y_{I_u} + y_c - y_{I_u} = y_c, \tag{11}$$

thanks to the consistency of our input estimates. Also, let $i = I_l$ and $j = J_l$ make the $(\tilde{y}_i + \tilde{z}_c^j - \tilde{z}_i^j)$ (in Eq. 9) the minimum. Then, a lower bound of the $\hat{y}_c$ can be computed:

$$\begin{aligned}\hat{y}_c &\ge \frac{1}{W_c}\sum_{j=1}^{m-1}\left[\sum_{i\in\Omega_c} w_i^j\left(\tilde{y}_{I_l} + \tilde{z}_c^{J_l} - \tilde{z}_{I_l}^{J_l}\right)\right] \\ &= \tilde{y}_{I_l} + \tilde{z}_c^{J_l} - \tilde{z}_{I_l}^{J_l}.\end{aligned} \tag{12}$$

Its limit $\lim_{N_{pass}\to\infty}\left(\tilde{y}_{I_l} + \tilde{z}_c^{J_l} - \tilde{z}_{I_l}^{J_l}\right) = y_{I_l} + y_c - y_{I_l} = y_c$. Note that the limits of both lower and upper bounds of the $\hat{y}_c$ go to the correct solution y_c. Consequently, $\lim_{N_{pass}\to\infty}\hat{y}_c = y_c$ by the squeeze theorem.