Eurographics Symposium on Rendering (DL-only Track) (2020)
C. Dachsbacher and M. Pharr (Editors)

Real-time Monte Carlo Denoising with the Neural Bilateral Grid

Xiaoxu Meng[1], Quan Zheng[1,2], Amitabh Varshney[1], Gurprit Singh[2], Matthias Zwicker[1]

[1]University of Maryland, College Park, USA
[2]Max Planck Institute for Informatics, Germany

Abstract

*Real-time denoising for Monte Carlo rendering remains a critical challenge with regard to the demanding requirements of both high fidelity and low computation time. In this paper, we propose a novel and practical deep learning approach to robustly denoise Monte Carlo images rendered at sampling rates as low as a single sample per pixel (1-*spp*). This causes severe noise, and previous techniques strongly compromise final quality to maintain real-time denoising speed. We develop an efficient convolutional neural network architecture to learn to denoise noisy inputs in a data-dependent bilateral space. Our neural network learns to generate a guide image for first splatting noisy data into the grid, and then slicing it to read out the denoised data. To seamlessly integrate bilateral grids into our trainable denoising pipeline, we leverage a differentiable bilateral grid, called neural bilateral grid, which enables end-to-end training. In addition, we also show how we can further improve denoising quality using a hierarchy of multi-scale bilateral grids. Our experimental results demonstrate that this approach can robustly denoise 1-*spp *noisy input images at real-time frame rates (a few milliseconds per frame). At such low sampling rates, our approach outperforms state-of-the-art techniques based on kernel prediction networks both in terms of quality and speed, and it leads to significantly improved quality compared to the state-of-the-art feature regression technique.*

CCS Concepts

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

1. Introduction

Monte Carlo ray tracing has become the *de facto* standard in movie production for its ability to accurately simulate physics-based global illumination. Large numbers of ray samples, however, have to be invested to obtain noise-free images. This leads to significant computational requirements that have restricted such techniques to offline rendering. Therefore, real-time, high-quality Monte Carlo ray tracing has been an elusive goal until now.

Currently, two main paths appear most promising to achieve fast and noise-free Monte Carlo rendering. The first path is to develop ever more powerful processors that are enhanced with ray tracing-specific functionality, like Nvidia's RT Cores [Wik18]. While recent progress has been impressive, high quality, real-time rendering without additional post-processing is unlikely to become practical in the near future. In addition, adapting existing ray tracing software frameworks to new hardware is not an easy task in itself. The second path is to generate noise-free images from noisy inputs rendered in real-time with very low numbers of ray samples. Since this approach can be implemented as a post-processing module without significant changes to existing renderers, it has gained overwhelming popularity in industrial production and academic research. The most powerful techniques from recent research on offline denoising rely on deep convolutional neural networks (CNNs). While Bako et al. [BVM*17] and Vogels et al. [VRM*18] developed denoising networks to predict pixel-centric kernels for filtering, Gharbi et al. [GLA*19] predict sample-centric kernels for splatting individual rendering samples to pixels. These techniques achieve good image quality, yet they have been designed for input images rendered with at least $8 \sim 16$ *spp*, which currently precludes them from real-time applications. Chaitanya et al. [CKS*17] proposed a denoiser for interactive applications that are targeted at images rendered with low sampling rates ($1 \sim 4$*spp*). In addition, this technique was later integrated into the *NVIDIA OptiX* engine and it can be considered as the current industry standard for interactive denoising. The approach is based on an autoencoder architecture that is simple enough such that interactive frame rates can be maintained. The limited capacity of the neural network, however, leads to compromised image quality compared to offline techniques. In this paper, we propose a novel deep learning approach to denoise Monte Carlo images rendered with extremely few samples. The core idea of our approach is to leverage the neural bilateral grid [GCB*17] for denoising. We splat noisy image data into a data-dependent bilateral grid, and the data-dependent mapping of image pixels into the grid is implemented using a neural network. The intuition is that effective data-dependent splatting into the bilateral grid can be learned with a simpler network than denoising the image itself, and denoising via the bilateral grid is very fast as its complexity does not depend on the support of the resulting denoising filter. In practice, our network predicts a *guide image* to

DOI: 10.2312/sr.20201133

control how image data is splatted into the bilateral grid. In the bilateral space, splatting samples are fused by simple tent filtering. Because of the low resolution of the grid, this effectively denoises the data. As a final step, we reuse the *guide image* from the neural network to extract data from the bilateral grid to reconstruct the denoised results, which is called *slicing*. Notably, we implement the bilateral grid as a differentiable component, such that we can seamlessly integrate it with the neural network for end-to-end training.

In addition, we propose a hierarchy of multi-scale bilateral grids that leads to further improvements of denoising quality at a slight increase of the computational cost. Finally, to denoise consecutive frames from a video, we apply temporal enhancement steps to ensure temporal stability. Inspired by [KIM*19], we apply temporal accumulation of consecutive frames in our pipeline to increase effective sampling rates and suppress temporal artifacts.

According to comprehensive experiments, our light-weight neural network leveraging the neural bilateral grid is able to denoise noisy input frames at real-time frame rates (a few milliseconds per frame), and at the same time to provide comparable and even higher quality for 1-*spp* data than the state-of-the-art real-time denoising approaches. In addition, with a deeper neural network architecture, we can further improve denoising quality while still ensuring interactive rates at dozens of frames per second. To summarize, our approach makes the following contributions:

- A novel, practical and robust denoising approach using neural networks. Experiments have demonstrated better denoising quality than existing methods at very low sample rates.
- A method for denoising using differentiable neural bilateral grids, which allow end-to-end training and efficient inference.
- A multi-scale pyramid of bilateral grids to process data at multiple levels and fuse them by an optimal weighted combination.

2. Related Work

Owing to its practicality and ease of use, Monte Carlo denoising has drawn extensive attention both in industrial productions and academic studies. This section focuses on research work that is most related to ours, and we refer to the existing literature for broader surveys [ZJL*15].

Offline Denoising. In offline denoising, a promising strategy is to derive sampling and reconstruction components based on the analysis of light transport, including frequency analysis [ODR09, ETH*09, BSS*13], light field structure analysis [LAC*11], or derivative analysis [BBS14]. These techniques have been categorized as "a priori" approaches, since they depend on tractable analytical analysis of the light transport equations. Inspired by the success of non-linear image filters, "a posteriori" approaches also flourish in Monte Carlo denoising. They achieve adaptive sampling and reconstruction by adapting non-linear image-processing filters in the image space, like cross-bilateral [LWC12] and non-local means filters [RKZ12, KS13]. Many methods [RKZ11] estimate parameters of the filters based on rendering errors and improve the estimation of parameters from auxiliary feature buffers [RMZ13], such as depth, albedo, roughness, and normals. As an alternative, local regression-based approaches [MCY14, BRM*16] seek the relationship between feature buffers and radiance and also achieve excellent denoising performance for off-line rendering.

Following the success of deep neural network in image processing, Kalantari et al. [KBS15] proposed a fully-connected neural network to estimate parameters of bilateral denoising filters. Bako et al. [BVM*17] and Vogels et al. [VRM*18] present a convolutional architecture to predict per-pixel filtering kernels, whereas Gharbi et al. [GLA*19] propose to predict kernels on a sample basis and splat samples onto pixels. In gradient-domain rendering, Kettunen et al. [KHL19] developed a convolutional neural network for the screened Poisson reconstruction process and achieve a significant quality improvement compared to the original gradient domain path tracing [KMA*15]. Recently, generative adversarial architectures [XZW*19] were also employed to denoise Monte Carlo rendering. While the above neural network-based approaches often achieve better denoising performance compared to traditional techniques, they may suffer from high computational costs caused by deep neural network architectures, which may limit them to offline rendering. In contrast, our neural approach can robustly denoise ill-posed noisy input (even with 1-*spp*) at real-time frame rates.

Real-time Denoising. Real-time path tracing reconstruction generally aims at 1-*spp* noisy input data to limit computation time, which requires powerful post-processing and denoising steps to obtain acceptable output images. Many approaches for fast filtering have been developed [Bur81, DSHL10, MMBJ17] for example based on hierarchical filtering, wavelets, or fast approximations of the bilateral filter [TM98]. A recent neural network approach [CKS*17] based on an encoder-decoder achieves interactive frame rates for 720p frames, while slightly compromising on quality compared to offline methods. Using the same architecture, two convolutional neural networks are employed by Kuznetsov et al. [KKR18] to jointly estimate sampling maps for adaptive sampling and remove Monte Carlo noise of 4-*spp* images.

Motivated by guided image filtering [HST10], fast reconstruction methods have also been proposed for Monte Carlo denoising. Bauszat et al. [BEM11, BEJM15] decompose light contributions into different components (e.g. direct and indirect lighting) and fits pixel filters for each component separately, achieving interactive frame rates. Along this line of work, Koskela et al. [KIM*19] propose to handle feature regression in a blockwise manner. By exploiting a data accumulation technique and augmented matrix factorization, this approach achieves reasonable image quality and runs at a real-time speed. Inspired by this work, we incorporate data accumulation into our neural bilateral grid denoiser. In general, our neural network based approach guarantees real-time denoising performance, and at the same time, it does not sacrifice local image details, which often plague regression based methods.

To ensure temporal stability of denoised frames, temporal accumulation [SKW*17] is adopted to increase effective rendering samples and reduce temporal errors. Based on the assumption of the coherence of consecutive frames, this technique can reuse temporal information via accumulation. To further reduce temporal artifacts, Schied et al. [SPD18] estimate temporal gradients and derive adaptive temporal accumulation factors. In neural network approaches [CKS*17], recurrent connections can also be adopted to alleviate temporal noise.

Bilateral Grid. Bilateral filtering [TM98] is a renowned nonlinear image processing technique for its advantages in smoothing images while preserving edges. Paris and Durand [PD06] reformulate bilateral filtering as fast linear filtering by lifting image data to a higher dimensional space. Chen et al. [CPD07] formally define the bilateral grid and present the edge-aware benefits of bilateral grids in typical image manipulation tasks. In the bilateral space, Chen et al. [CAWH16] fit affine models to approximate image processing operators and achieve efficient image manipulation for high-resolution images. The bilateral grid has been generalized to solve colorization and semantic segmentation problems [BP16].

In addition, Gharbi et al. [GCB*17] train a neural network to learn affine color transforms to conduct image enhancement. They embed the low-resolution input as coefficients in the bilateral space and use the sliced coefficients to perform a pointwise transformation. Motivated by the edge-aware advantage of the bilateral grid, we incorporate the neural bilateral grid in our denoising pipeline. Different from Gharbi et al. [GCB*17], we embed the noisy radiance data in the bilateral grid by accumulating multiple neighboring pixels into a specific grid element, and slice directly from the grid to get the denoised data. Our approach avoids the explicit computation of per-pixel weights for large kernels and achieves high-quality denoising with fast, spatially uniform filters.

3. Problem statement

Our goal is to learn a denoiser to reconstruct noise-free images from noisy rendering data. With existing full-fledged renderers and typical scene description data, we can generate a large number of noisy images rendered with k samples-per-pixel (k *spp*) and their approximate noise-free versions (4096 *spp*). In addition to noisy images, auxiliary features like normal, depth, and albedo can readily be obtained as by-products from the rendering process. We generate a set of paired data of N frames $\{((r_1,\mathbf{f}_1),t_1), ((r_2,\mathbf{f}_2),t_2), \ldots, ((r_N,\mathbf{f}_N),t_N)\}$, where r_i, $\mathbf{f}_i$ stands for noisy radiance and auxiliary features of frame i, and t_i is the noise-free ground truth of frame i. Given the dataset, we formulate the denoising problem as a supervised learning task. In this paper, we propose a novel architecture which composites a neural network and neural bilateral grids to solve the problem.

The traditional bilateral grid is designed as a three-dimensional data structure [CPD07], although generalizations to higher dimensional structures are possible. To process an image with a bilateral grid, the image is firstly mapped to the 3D bilateral grid, then the bilateral grid is processed according to the specific image processing task, and finally the output is obtained by extracting a 2D slice of the 3D grid.

In a similar spirit, we design a neural bilateral grid for denoising Monte Carlo renderings. A bilateral grid M_i is a three dimensional embedding of an image, which we can write as $M_i : \mathbb{R}^3 \to \mathbb{R}^3$, where the input is a location in 3D space, and the output is a radiance value. First, we set up a bilateral grid by embedding noisy radiance data into the grid. Let us denote the j-th pixel in a noisy input frame as $r_i(j)$. We denote the construction of a bilateral grid for frame i as

$$M_i(p) = \Phi\{r_i|g_i\}(p) = \sum_j r_i(j)\delta(g_i(j)-p), \tag{1}$$

where $p \in \mathbb{R}^3$ is a spatial location, δ is a Dirac impulse, and $g_i(j) \in \mathbb{R}^3$ is a guide which indicates the desirable placement of noisy radiance data of each pixel in the bilateral grid.

Then, we apply a three-dimensional filter $\mathcal{F}$ to M_i to achieve a specific image processing task, that is, noise removal from Monte Carlo renderings in our application. Our filtering process is a simple convolution of the filter $\mathcal{F}$ with M_i,

$$M_i'(p) = M_i \circledast \mathcal{F}(p) = \sum_j \mathcal{F}(g_i(j)-p)r_i(j), \tag{2}$$

where $\circledast$ is a three dimensional convolution. This equation can be interpreted as splatting each noisy pixel $r_i(j)$ into the grid using a filter kernel $\mathcal{F}(g_i(j)-p)$ that is centered at the position $g_i(j)$ given by the guide.

For the next step, we reconstruct frame i by extracting data from the processed bilateral grid M_i'. This operation is called *slicing*,

$$R_i(j) = M_i'(g_i(j)), \tag{3}$$

where R_i is the reconstructed image, $R_i(j)$ is its j-th pixel, and slicing is performed by reading the grid at the 3D location $g_i(j)$ using the same guide as for grid construction.

By chaining the above three operations together, we define the denoising process as a unified operation

$$R_i(j) = \{\Phi\{r_i|g_i\} \circledast \mathcal{F}\}(g_i(j)) \tag{4}$$

which suggests an end-to-end denoising process from input noisy radiance to denoised results.

A key idea in our approach is that for a suitable guide g_i we can use a very simple and fast convolution operation F. Hence, instead of manually crafting a guide, we propose to use a neural network $\mathcal{C}$ to predict a guide. In addition to noisy radiance r_i, we also feed available auxiliary features $\mathbf{f}_i$ to the neural network. This process can be expressed as

$$g_i(j) = (j_x, j_y, \mathcal{C}_\Theta(r_i,\mathbf{f}_i)(j)), \tag{5}$$

where Θ represents trainable parameters of our neural network. Note that for simplicity, the pixel coordinates j_x, j_y are re-used as the first two coordinates of the guide, and the neural network only predicts the third coordinate. Hence the neural network output $\mathcal{C}_\Theta(r_i,\mathbf{f}_i)$ is an effective 2D guide image.

By substituting the above form of g_i into Equation 4, we can further rewrite the denoising operation as

$$R_i(j) = \{\Phi\{r_i|\mathcal{C}_\Theta(r_i,\mathbf{f}_i)\} \circledast F\}(j_x, j_y, \mathcal{C}_\Theta(r_i,\mathbf{f}_i)(j)). \tag{6}$$

Intuitively, we drive the neural network to learn to place noisy radiance data in a bilateral grid in an optimal way. The neural network is trained in an end-to-end manner based on the training data. We define a loss function $\mathcal{L}$ to measure the difference between one denoised image R_i and its ground truth image t_i. To obtain the optimal parameters Θ_{opt} of the neural network, we minimize the loss

function with modern gradient descent algorithm, given a set of k training samples,

$$\Theta_{opt} = \arg\min_{\Theta} \sum_{i=1}^{K} \mathcal{L}(R_i, t_i). \quad (7)$$

4. Neural Bilateral Grid Denoiser

In this section, we introduce the overall multi-scale architecture of our neural bilateral grid denoiser. As shown in Figure 1, the input data consist of path-traced noisy radiance and its auxiliary features. Input data are sent to a *GuideNet* to predict *guide* images (Section 4.1). Following that, *guide* images instruct the construction of the bilateral grid (Section 4.2). In the next stage, we reuse *guide* images to slice data from the bilateral grid and reconstruct a denoised image (Section 4.3). Lastly, we describe how to train the *GuideNet* in an end-to-end fashion based on the complete pipeline including grid construction and slicing (Section 4.4).

In addition, we design a multi-scale architecture (Section 4.5) with three bilateral grids of different resolutions. At each scale, the same construction and slicing on a bilateral grid are executed independently. The final output is a weighted combination of results from multiple scales. Finally, we include temporal enhancement operations (Section 4.6) to ensure temporal stability when denoising image sequences.

4.1. Guide Prediction

We design the *GuideNet* module in Figure 1 with a convolutional neural network (CNN) architecture. For denoising 1-*spp* input at real-time speed, we use a two-layer CNN (Figure 2, left). For denoising 64-*spp* input that are not produced in real-time, we utilize a seven-layer CNN architecture that contains a five-layer dense block (Figure 2, right). For each layer in the block, we concatenate their feature maps to its following layers. In both architectures, we set the 2D size of convolution as 5×5. We employ rectified linear (ReLU) [NH10] activations for layers before the last one, while we use Sigmoid activation for the last layer.

Data decomposition. As a preprocessing step, we factor out albedo from the noisy radiance data. Symmetrically, we multiply the albedo back to the denoised radiance data at the end. By removing albedo, the neural network can focus on learning to predict a guide based on noise patterns in smooth irradiance data. The decomposition and composition of albedo requires only one division and one multiplication, incurring negligible computational costs.

4.2. Bilateral Grid Construction

The bilateral grid in our framework is a three-dimensional grid with a resolution $H \times W \times D$ (Figure 1). To splat a pixel with RGB color channels onto the grid, we leverage the corresponding scalar value in the guide image to instruct the placement of the three-dimensional spectrum vector as a whole.

To map a two-dimensional image to a bilateral grid, we define three sampling rates η_h, η_w and η_d for the three axes of the grid. The resolution of a bilateral grid is related to the sampling rates by

$$(H, W, D) = \left(\lceil \frac{h}{\eta_h} \rceil, \lceil \frac{w}{\eta_w} \rceil, \lceil \frac{d}{\eta_d} \rceil \right), \quad (8)$$

where h and w are the height and width of the two-dimensional noisy image, and d is the range of the guide, which we set to 255.

We then directly obtain a filtered, discrete grid by computing the filtered grid M_i' as in Equation 2 and evaluating it at the discrete grid points. For simplicity, we use a tent filter

$$T(x,y,z) = \max(1-|x|,0) \cdot \max(1-|y|,0) \cdot \max(1-|z|,0) \quad (9)$$

to splat the noisy data into the grid, that is $\mathcal{F} = T$ in Equation 2. The definition of T assumes unit spacing between grid cells. To avoid the singularity in the derivative of the absolute value function $|\cdot|$, we replace $|a|$ in T with $\sqrt{a^2+\varepsilon}$, where a can be x, y or z and ε is a small number $1e-7$. The function T now provides non-zero gradients for points within its support, which is what we desire.

Grid cells in a bilateral grid M_i' are initialized with zero. Given the tent function as the splatting filter, the noisy value $r_i(j)$ of pixel j can be added to the grid cell simply by an equivalent tri-linear interpolation. Following Equation 2, for a grid point p with coordinates $p = (x,y,z)$, its stored value is

$$M_i'(p) = \frac{\sum_j w_{p,j} \cdot r_i(j)}{\sum_j w_{p,j}}, \quad (10)$$

where the sum is over all pixels j of the noisy image R_i. The normalization by the sum of the weights $\sum_j w_{p,j}$ is necessary because of the nonuniform distribution of splatting kernels in the bilateral grid. The weights $w_{p,j}$ for each pair of grid point p and pixel j are

$$w_{p=(x,y,z),j=(m,n)} = T(g_i(j) - p), \quad (11)$$

where for a pixel $j = (m,n)$ on the noisy frame i, its position in the bilateral grid is given by the guide $g_i(j)$ as

$$g_i(j) = (m/\eta_h, n/\eta_w, \mathcal{C}_\Theta(r_i, \mathbf{f}_i)(j)/\eta_d). \quad (12)$$

In practice, pixels whose weights $w_{p,j}$ are guaranteed to be zero can be excluded from Equation 10.

4.3. Bilateral Grid Slicing

We reconstruct a denoised image by *slicing* the bilateral grid. This is a semantically symmetrical procedure to the bilateral grid construction. With a guide g_i, reconstructing a pixel j involves evaluating $M_i'(g_i(j))$ as formulated in Equation 3.

Since our grid M_i' is discrete, this requires continuous interpolation of the grid values. Similar as for splatting noisy data into the grid described by Equation 10, we employ tri-linear interpolation using the tent filter T to interpolate values from the bilateral grid.

To compute a pixel j in the sliced image $R_i(j)$, we return a weighted combination of grid values based on the interpolation filter. By splatting a 2D noisy image to a 3D bilateral grid, the bilateral grid is sparsely populated and some of its cells remain empty.

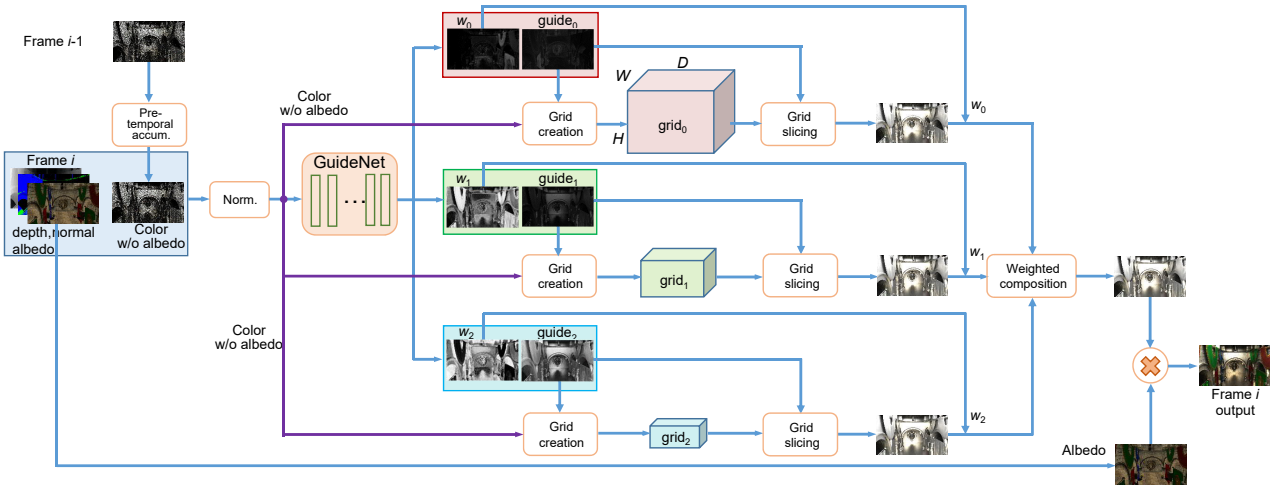

Figure 1: *Overview of our temporal, multi-scale neural bilateral grid denoiser. The temporally accumulated noisy radiance without albedo, and auxiliary features including depth, normals, and albedo of each frame are sent to a GuideNet. GuideNet is trained in an end-to-end fashion to embed the noisy image data in the bilateral grid, and to slice radiance from the bilateral grid. In addition, we apply a multi-resolution approach using grids at three resolutions, and GuideNet also provides blending weights to combine their outputs. Finally, we multiply the albedo back to form a denoised frame.*

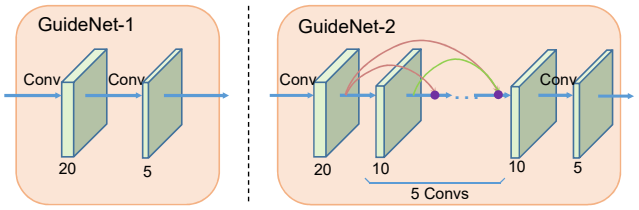

Figure 2: *Architectures of GuideNet. For 1-*spp *input, a two-layer CNN architecture (left) is used. For 64-*spp *input, a seven-layer architecture (right) with a dense block is employed. The dense block contains five layers and the purple dots are concatenations. The number of filters of each layer are shown below each layer. The output consists of three guide images and two pixel-wise weights.*

That is, these cells did not receive any contributions during splatting. We skip empty grid cells with zero values and include a renormalization,

$$R_i(j) = \frac{\sum_{\{p|M_i'(p)\neq 0\}} M_i'(p)\cdot w_{p,j}}{\sum_{\{p|M_i'(p)\neq 0\}} w_{p,j}}, \tag{13}$$

where $\{p|M_i'(p)\neq 0\}$ indicates the set of integer grid points where the grid is non-zero. The weights $w_{p,j}$ are computed identically to the splatting weights in Equation 11, using the same tent filter T and guide $g_i(j)$. In practice, the summation in Equation 13 can be limited to grid cells within the support of the interpolation filter centered at $g_i(j)$. Note that in general the splatting filter does not need to be the same as the interpolation filter for slicing. In addition, we could also use two different guides for splatting and slicing. In our experiments, however, we did not observe any benefits of this.

Also, we could set up another neural network to produce such a guide $g_{\Theta'}$. We, however, do not observe significant improvement of denoising performance by computing two independent guides. Hence, we employ only one neural network and reuse the predicted guide for both construction and slicing of a bilateral grid.

Post-processing. The denoised images from the *slicing* step are irradiance data without albedo. As a post-processing, we multiply the albedo data in the auxiliary features back to the denoised result R_i, and obtain an output $\tilde{R}_i = R_i * \mathbf{f}_{\text{albedo}}$.

4.4. End-to-end Training

We train our *GuideNet* that we use to predict the guide to minimize the objective as formulated in Equation 7. We use the $L1$ loss function, which computes the mean absolute difference between denoised results and ground truth images, since it correlates well with the difference perceived by humans. Hence, our loss is

$$\mathcal{L} = \frac{1}{K}\sum_{i=1}^{K} \left|\mathcal{S}\left(\tilde{R}_i\right) - \mathcal{S}\left(t_i\right)\right|. \tag{14}$$

Here, t_i represents a ground truth image. $\mathcal{S}$ represents an image processing function associated with the training dataset (details are in Section 5). The input noisy radiance data are processed by $\mathcal{S}$ before entering the neural network. Note that we use high dynamic range data in the bilateral grid construction and slicing modules to preserve information of the original dynamic range.

In our pipeline, bilateral grid construction and bilateral grid slicing modules are inserted between the neural network and the final output. To allow error back-propagation from the final output to the neural network, we derive analytical gradients of the two modules. Specifically, the gradient of bilateral grid with respect to the guide

$\frac{\partial M}{\partial g}$ is computed for constructing the bilateral grid M. For the slicing module, we compute gradients of the denoised output with respect to the grid and the guide, $\frac{\partial R}{\partial M}$ and $\frac{\partial R}{\partial g}$. For details of the analytical derivatives, we refer readers to our supplementary material.

4.5. Multi-scale Bilateral Grids

Given the input noisy frame, we build a 3-scale pyramid of bilateral grids as shown in Figure 1. We denote the scales using index $s \in \{0,1,2\}$. As in Section 4.2, the resolution of a grid of scale s is given by sampling rates $(\eta_h^s, \eta_w^s, \eta_d^s)$. The bilateral grid at scale 0 has the original, highest resolution $(\eta_h^0, \eta_w^0, \eta_d^0) = (\eta_h, \eta_w, \eta_d)$. For lower scales, we use $(\eta_h^{s+1}, \eta_w^{s+1}, \eta_d^{s+1}) = (2\eta_h^s, 2\eta_w^s, 2\eta_d^s)$. That is, the following scale will halve the resolution of the previous scale along every direction of the grid.

For each scale s, we generate a separate guide image from the *GuideNet*, and use it to instruct the construction of the bilateral grid with the appropriate resolution. Similarly, we *slice* each bilateral grid scale with its individual guide image and obtain reconstructed radiance R_i^s independently. To form the final radiance, we compute the weighted composition

$$R_i = w_0 \cdot R_i^0 + w_1 \cdot R_i^1 + w_2 \cdot R_i^2, \tag{15}$$

where w_0, w_1, w_2 are pixel-wise weight maps. We reuse the *GuideNet* to predict weight maps $0 \le w_0 \le 1$, $0 \le w_1 \le 1$ and set $w_2 = 1 - (w_0 + w_1)$ to normalize the weights.

4.6. Temporal Enhancement

When denoising consecutive frames of a video, temporal stability is important for perceived visual quality because the random residual, low frequency noise patterns vary from frame to frame. To improve temporal stability, we utilize a pre-stage of temporal enhancement to accumulate consecutive 1-*spp* noisy radiance data without albedo (Figure 1) by the reprojection technique [KIM*19, SKW*17], where we set the decay factor of the previous frame to 0.2.

5. Experimental Setup

Datasets Preparation. We used two datasets in our experiments. We firstly adopt an existing dataset from the work of Koskela et al. [KIM*19], which we call the BMFR dataset. The dataset is comprised of a set of 360 noisy 1-*spp* rendering images and their ground truth images from six scenes: *Classroom*, *Living-room*, *San-miguel*, *Sponza*, *Sponza-glossy*, *Sponza-moving-light*. These scenes are configured with diverse materials, illumination and camera movements. Each scene has 60 consecutive frames and their ground truth images are rendered with more than 2048 *spp* to obtain approximate noise-free appearance. The image resolution is 1280×720.

In addition, we build a new dataset of publicly available Tungsten scenes [Bit16] using the Tungsten renderer and we call it Tungsten dataset. Our dataset includes rendered images at a resolution of 1280×720 from nine scenes, which cover a wide variety of geometry scales, material types, and lighting effects that reflect typical challenges for modern renderers. See Figure 3 for an overview of scenes used in our dataset. For every scene, we rendered a consecutive sequence of 100 frames. For each frame, we use the built-in path tracer of Tungsten with its default settings to render 1 and 64-*spp* noisy images. The corresponding ground truth images are rendered with 4096 *spp*.

Figure 3: *An overview of reference frames of scenes included in our own dataset.*

From the noisy images, we noticed visible outlier pixels whose values are significantly higher than their neighbors. Splatting outliers with extremely high value to the bilateral grid will cause artifacts since these values cannot be effectively spread. To reduce the impact of outliers, we apply a preprocessing step [KBS15] to detect and suppress outlier pixels.

The *GuideNet* accepts images with low dynamic range, so we use the function $\mathcal{S}$ (Equation 14) to convert input noisy radiance data to low dynamic range data. $\mathcal{S}$ represents the gamma correction function for BMFR dataset with exponent of $1/2.2$, whereas for the Tungsten dataset it represents the Tungsten tone mapping operator [Bit16].

Training and Testing. For both datasets we hold out frames of one scene as test data, and use the remaining images of the other scenes as the training data. In addition, we randomly select frames of one scene from the training data to function as validation data. This means we use a separate network to obtain test results for each scene, where the network is trained using data only from the other scenes. Hence the performance on test data can serve as an indication of the generalization ability of the trained denoiser.

During both the training and the inference stage, inputs to the *GuideNet* are comprised of low dynamic range (LDR) radiance with albedo (3 channels) and auxiliary features (7 channels), which include 3-channel albedo, 3-channel normal and 1-channel depth. The input to the bilateral grid is the high dynamic range (HDR) radiance without albedo. Both the radiance and auxiliary features are normalized to the range between 0 and 1 before sending to the network. In the training stage, we use image patches at a resolution of 128×128. We randomly sample 60 such crops from each frame of training data to form a training dataset, and sample 20 crops to build a validation dataset. In the inference stage, the inputs to the network consist of the full resolution (1280×720) frames from the test data.

Implementation. Our denoiser is implemented in TensorFlow [AAB*15]. The bilateral grid construction and slicing modules have been implemented in CUDA as plug-in operators to integrate with TensorFlow. Before training, weights of the neural network are initialized with uniform random value ranges from -1 to 1. Three sampling rates for the bilateral grid at scale-0 are configured as $\eta_h = \eta_w = 4$, $\eta_d = 4$. Depending on the GPU memory size, we use mini-batches of size 20 in training. Because the full frames for testing are larger than the crops for training, we set the mini-batch size to 10 for testing. We train the pipeline in an end-to-end fashion using Adam [KB14]. We set the learning rates to 0.0001 and keep the default values for the other parameters. A typical training time for one test scene using 100 epochs on an Nvidia RTX 2080 GPU is about 14 hours.

6. Results and Discussion

We evaluate our approach by comparing it with state-of-the-art methods in terms of both image quality and running speed.

6.1. Evaluation Methods and Error Metrics

Since our approach aims at real-time denoising, we mainly focus our comparisons on existing real-time denoising approaches. Specifically, we report results from the recent BMFR approach [KIM*19], the OptiX Neural Network Denoiser (ONND) which is adapted from [CKS*17] and is available in the OptiX 5.1 engine, Spatio-temporal Variance-Guided Filtering (SVGF) [SKW*17], and a Multi-Resolution variant of Kernel Prediction CNN (MR-KP) denoiser [BVM*17]. In addition, we also include a comparison with an off-line denoiser, NFOR [BRM*16]. For BMFR, SVGF and NFOR, we use implementations provided by their authors.

The original kernel prediction denoiser [BVM*17] is designed for off-line denoising and requires several seconds to process a single 720p frame. Similar to our multi-scale design, we have adapted the multi-resolution variant (MR-KP) [VRM*18], which decreases the run time of a basic kernel prediction architecture to the order of tens of milliseconds. Hence it is an important comparison point for real-time denoising. Details of the architecture of MR-KP can be found in the supplementary material. For fair comparison, our MR-KP is trained with the same loss and the same split of dataset as our approach. Also, we perform the same temporal filtering as in our approach, instead of including it in the kernel prediction denoiser itself.

ONND is provided as a black box module in OptiX 5.1. It is derived from the autoencoder-based denoiser by Chaitanya et al. [CKS*17], but ONND differs from the original paper in several aspects: 1) ONND consumes a different set of auxiliary features. 2) It expects low dynamic range input which has been processed by tone mapping and gamma correction and it outputs low dynamic range data.

Since ONND operates albedo demodulation and re-modulation in its black box, we send original noisy radiance to it and skip the re-modulation of albedo. For BMFR, SVGF and MR-KP, we apply the same albedo demodulation and re-modulation processes as in our approach. Note also in NFOR, the albedo processing is not applied. To evaluate denoising performance quantitatively, we adopt standard metrics including PSNR and SSIM [WSB03] (see also error maps in the supplementary material).

Figure 4: *Scatter plots of PSNR-timing and SSIM-timing of all evaluated denoisers.*

6.2. Quality Comparisons

Our key objective is to achieve very fast denoising for low quality data such as 1-*spp* path tracing. In this scenario, we use a light-weight 2-layer convolutional neural network as the *GuideNet* and three relatively coarser bilateral grids to strike the balance between fast speed and good quality. We demonstrate that our approach can outperform the previous state of the art in this scenario.

Owing to the scalable and modular architecture of our denoiser, we can tailor the architecture of the *GuideNet* and adjust the resolution of bilateral grids to adapt to different applications. For 64-*spp* data, which are naturally produced with offline settings, we have employed a more complex *GuideNet*, which is designed as a 7-layer DenseNet [HLVDMW17]. We demonstrate competitive performance in this setting, although denoising data from off-line rendering is not intended as the main application of our approach.

For 1-*spp* input data to our denoiser, we apply pre-stage temporal accumulation to increase the effective sample count of each frame as described in Section 4.6. To ensure fairness, we use the same temporal accumulation for all compared approaches. Our denoiser is applicable to reconstruct both single noisy images and a sequence of noisy animated frames. In the following sections, we show denoised results of single frames. Besides, we upload a test suite with an interactive viewer along with this paper to allow readers to view and compare all denoised frames.

BMFR dataset. Firstly, we show how our denoiser performs on the BMFR dataset. As described in Section 5, we use frames of five scenes as the training dataset, and test the denoiser on frames of the remaining one scene. Figure 5 shows the comparison of the 51-th

frames of the selected test scene. Our method performs better at preserving regions with highlights, as shown in the closeup on the lion relief of the *Sponza (glossy)* scene. The error measurements are presented in Table 1 and Table 2. It can be observed that our method generally provides low numerical errors. Also, scatter plots in Figure 4 clearly show that our denoiser performs the best in terms of PSNR and SSIM, meanwhile our denoiser stands a comparable position for speed.

To measure the time cost of denoising, we apply all denoisers on the same test data. The mean time cost to denoise a frame is averaged over all test frames. The average timings of all methods are presented in Table 3. For denoising the BMFR dataset, our denoiser using a 2-layer *GuideNet* and a 3-scale bilateral grids (2nd row of Table 3) can run at a real-time performance which reaches 61 frames per second (FPS). Our method does not show obvious superiority in denoising quality compared with 5-layer MR-KP, but ours (16.460 ms) is faster than the 5-layer MR-KP (39.537 ms). If we reduce the capacity of MR-KP to make it faster by going to one layer, it still ends up with slower speed and lower denoising quality compared to ours.

Tungsten dataset. For our Tungsten dataset, we rendered noisy image sequences at 64-*spp*. Figure 6 shows the comparisons of denoised results of test scenes extracted from our dataset. Our test scenes contain a wide range of challenges for existing rendering techniques, including difficult glossy and specular light transport and complex visibility issues. Although rendered with 4096 *spp*, the reference images still show some residual noise as in the *Living Room* scene and the *Dining Room* scene. In our experiments, input data with and without removing outliers are tested. Experiments show that outlier removal leads to more visually pleasing results, thus we adopt these results in the following comparisons.

As shown in the Living Room scene (row 1 of Figure 6), our approach excels at preserving edges of the lights on the ceiling and edges of thin tree branches with slender structure, while NFOR and BMFR slightly blur the edges. Due to the blockwise reconstruction manner, BMFR will remove geometry details in some local regions.

Also, our denoiser can robustly reconstruct the shadow cast by the shutter blinds in the *Dining Room* scene (row 2 of Figure 6), whereas BMFR misses the regular structure of shadow. NFOR, which utilizes more sophisticated auxiliary features, can reconstruct the shadow with the highest quality, but it also leads to a much slower speed. The blue insets of the *Dining Room* scene show surfaces with glossy materials and highlights. Our denoiser, ONND and MR-KP effectively removes the noise and preserves the highlight regions, whereas BMFR cannot recover highlight and glossy reflections. While NFOR shows the highest quality of shadow, it slightly blurs the regions with specular highlights.

Table 4 presents the average errors over 100 frames on five scenes of our Tungsten dataset. Note that our method with outlier removal leads to a slight drop of PSNR and SSIM. To avoid affecting error metrics of other denoisers, we do not apply outlier removal to them. In our experiment, we exploit the outlier removal step because it can quench outliers and achieve visually better results.

As stated before, we use a *GuideNet* with a 7-layer architecture (Figure 2, right) for denoising 64-*spp* input data. We measure the average time cost by denoising all frames of the *Dining Room* scene of our Tungsten dataset. The average timing for denoising one 720p frame is shown in the third data row of Table 3.

Our denoiser in this case runs at 11.7 FPS. Since 64-*spp* renderings execute in offline applications, our denoiser can be applied as an efficient post-processing process to improve quality of rendered images.

6.3. Ablation Studies

Guide Prediction. To analyze the impact of training a neural network to obtain optimized guide images, we compare our approach to a baseline where the guide image is simply the arithmetic mean of the auxiliary features. Figure 7 compares our optimized guide images and the baseline on the *Sponza* scene. The baseline shows apparent artifacts around edges because it cannot guide a correct splatting of noisy radiance. On the contrary, our approach learns optimal guide images to place noisy radiance in the grid and gives better reconstructed results.

We further visualize the three guide images of three scales and the single guide image of the baseline in Figure 8. As shown in the closeups, our guides successfully distinguish the smooth dim region from the sharp edge of a rod, while the baseline is affected by multiple features and leads to artifacts in its output.

Multi-scale bilateral grids. Our denoiser uses a three-scale hierarchy of bilateral grids. The final result is reconstructed by a weighted blending of denoised images from the three scales. Based on the experiments on the test scenes presented in Figure 5, we calculate the average PSNR for denoised images at each scale. Then, we compare the arithmetic mean of three PSNRs with the PSNR of the final result produced by our weighted blending. As summarized in Table 5, the weighted multi-scale approach consistently produces higher quality than any of the individual scales.

The supplementary document includes additional ablation studies to compare different architectures of our denoiser.

6.4. Limitations

Specular light transport. Specular light paths are notoriously difficult to sample, thus often leading to extreme noise. In this scenario, features such as albedo textures also provide little useful information to the denoiser. The blue insets of the Living Room scene in Figure 6 shows such a case, where all of the denoisers fail at reconstructing the glass vase. Accurately denoising such difficult scenarios remains a challenge for real-time denoisers.

Generalization to unseen effects. Our scalable neural bilateral grid denoiser is able to provide better visual quality and lower residual error on the 1-*spp* BMFR dataset and competitive performance on the 64 *spp* Tungsten dataset, but it has several generalization issues. Since none of the datasets include scenes with depth of field, motion blur, or volumetric media, our denoiser may yield poor performance or give artifacts on such scenes with unseen effects. This problem can currently only be mitigated by increasing the generality of training scenes.

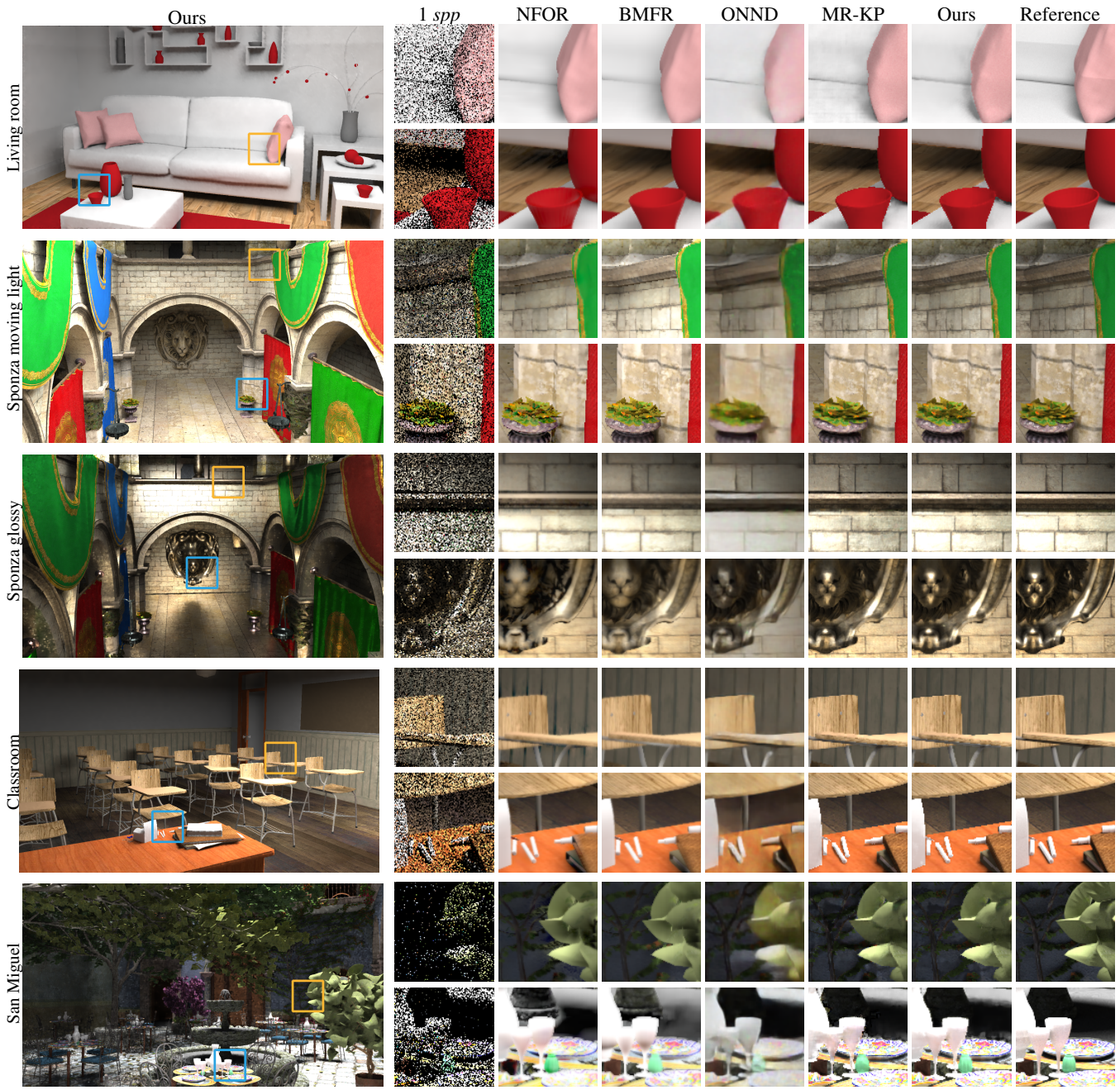

Figure 5: *Comparisons on the 1-*spp *BMFR dataset showing the* 51*-th frame from the animated sequence. Each row represents an independent experiment where the displayed scene is for test and other scenes are used for training. References are rendered with 4096* spp*. For each scene, closeups of the orange square are shown on the top row and closeups of the blue square are on the bottom row.*

7. Conclusions

We have introduced a novel and practical neural bilateral grid denoiser, that can reconstruct extremely noisy rendering images in real-time. At the core of our approach, we utilize an efficient neural network, *GuideNet*, to learn to splat noisy radiance data onto a hierarchy of multi-scale bilateral grids and then slice the grid to extract denoised radiance. The proposed neural bilateral grid denoiser is scalable and it allows us to tailor the design of each module according to real-time or offline applications. Our experiments have demonstrated that, with a two-layer shallow *GuideNet*, it is able to deliver better denoised results in terms of both visual quality and numerical error metrics for 1-*spp* input data. By using a deeper

Table 1: *A comparison of average PSNR values (higher is better) for evaluating our trained denoisers on 1-*spp *BMFR test data (see supplementary document for other error metrics). MR-KP 5-layer represents the 5-layer kernel prediction architecture and MR-KP 1-layer represents the 1-layer kernel prediction architecture. Ours 3-grid represents the 2-layer, 3-grid architecture and Ours 1-grid stands for the 2-layer, 1-grid architecture. We achieve the highest scores except for PSNR of Classroom and Sponza (mov. light), while taking less than ten milliseconds (see Table 3).*

Scene	PSNR							
	NFOR	BMFR	ONND	SVGF	MR-KP(5-layer)	MR-KP(1-layer)	Ours(3-grid)	Ours(1-grid)
Classroom	29.872	28.965	27.312	25.034	**32.177**	31.392	31.519	31.284
Living room	31.304	30.025	25.586	27.239	31.995	30.311	**32.294**	28.317
San Miguel	21.811	20.969	20.172	18.736	22.950	22.482	23.650	**23.940**
Sponza	30.377	31.111	24.698	24.401	29.476	29.912	**33.188**	32.787
Sponza (glossy)	25.974	25.005	23.460	20.917	26.447	27.303	**29.548**	29.459
Sponza (mov. light)	21.999	17.377	22.291	17.260	**25.090**	23.637	24.818	24.860

Table 2: *A comparison of average SSIM values (higher is better) for evaluating our trained denoisers on 1-*spp *BMFR test data (see supplementary document for other error metrics). MR-KP 5-layer represents the 5-layer architecture and MR-KP 1-layer represents the 1-layer architecture. Ours 3-grid represents the 2-layer, 3-grid architecture and Ours 1-grid stands for the 2-layer, 1-grid architecture).*

Scene	SSIM							
	NFOR	BMFR	ONND	SVGF	MR-KP(5-layer)	MR-KP(1-layer)	Ours(3-grid)	Ours(1-grid)
Classroom	0.956	0.955	0.924	0.952	**0.972**	0.959	0.968	0.966
Living room	0.967	0.965	0.953	0.950	0.965	0.933	**0.968**	0.948
San Miguel	0.799	0.789	0.744	0.790	0.808	0.773	0.820	**0.834**
Sponza	0.939	0.948	0.852	0.927	0.963	0.955	**0.973**	0.973
Sponza (glossy)	0.901	0.907	0.867	0.913	0.927	0.897	0.941	**0.943**
Sponza (mov. light)	0.895	0.858	0.811	0.876	**0.948**	0.896	0.946	0.947

Table 3: *Average time cost of each denoising approach. We separately record timings on the Classroom scene of two datasets. For the BMFR dataset, the timing is averaged over 60 Frames, while the timing is averaged over 100 frames for the Tungsten dataset.*

Method	Timing (ms)	Device
Ours 2-layer 1-grid	9.407	Nvidia RTX 2080
Ours 2-layer 3-grid	16.460	Nvidia RTX 2080
Ours 7-layer 3-grid	84.988	Nvidia RTX 2080
NFOR	370.000	Intel Core i7-8700 CPU
BMFR	1.600	Nvidia RTX 2080
ONND	55.000	Nvidia Titan X
SVGF	4.400	Nvidia Titan X
MR-KP(5-layer)	39.527	Nvidia RTX 2080
MR-KP(1-layer)	25.321	Nvidia RTX 2080

neural network architecture, its denoising ability is on par with or outperforms state-of-the-art real-time denoising approaches.

References

[AAB*15] ABADI M., AGARWAL A., BARHAM P., BREVDO E., ET AL.: TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software available from tensorflow.org. URL: https://www.tensorflow.org/. 7

[BBS14] BELCOUR L., BALA K., SOLER C.: A local frequency analysis of light scattering and absorption. *ACM Transactions on Graphics (TOG) 33*, 5 (2014), 163. 2

[BEJM15] BAUSZAT P., EISEMANN M., JOHN S., MAGNOR M.: Sample-based manifold filtering for interactive global illumination and depth of field. In *Computer Graphics Forum* (2015), vol. 34, Wiley Online Library, pp. 265–276. 2

[BEM11] BAUSZAT P., EISEMANN M., MAGNOR M.: Guided image filtering for interactive high-quality global illumination. In *Computer Graphics Forum* (2011), vol. 30, Wiley Online Library, pp. 1361–1368. 2

[Bit16] BITTERLI B.: Rendering resources, 2016. https://benedikt-bitterli.me/resources/. 6

[BP16] BARRON J. T., POOLE B.: The fast bilateral solver. In *European Conference on Computer Vision* (2016), Springer, pp. 617–632. 3

[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. In *Computer Graphics Forum* (2016), vol. 35, Wiley Online Library, pp. 107–117. 2, 7

[BSS*13] BELCOUR L., SOLER C., SUBR K., HOLZSCHUCH N., DURAND F.: 5d covariance tracing for efficient defocus and motion blur. *ACM Transactions on Graphics (TOG) 32*, 3 (2013), 31. 2

[Bur81] BURT P. J.: Fast filter transform for image processing. *Computer graphics and image processing 16*, 1 (1981), 20–51. 2

[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 Transactions on Graphics (TOG) 36*, 4 (2017), 97. 1, 2, 7

[CAWH16] CHEN J., ADAMS A., WADHWA N., HASINOFF S. W.: Bilateral guided upsampling. *ACM Transactions on Graphics (TOG) 35*, 6 (2016), 203. 3

[CKS*17] CHAITANYA C. R. A., KAPLANYAN A. S., SCHIED C., SALVI M., LEFOHN A., NOWROUZEZAHRAI D., AILA T.: Interactive reconstruction of monte carlo image sequences using a recurrent denoising autoencoder. *ACM Transactions on Graphics (TOG) 36*, 4 (2017), 98. 1, 2, 7

Table 4: *Error measurements for our trained denoisers on test data of the Tungsten dataset, which is rendered at 64* spp *and does not use temporal accumulation as a pre-process. 'Ours' denotes our denoiser with outlier removal preprocessing, whereas 'Ours wo' is without outlier removal. We use the 7-layer 3-grid architecture.*

Scene	PSNR						SSIM					
	NFOR	BMFR	ONND	MR-KP	Ours wo	Ours	NFOR	BMFR	ONND	MR-KP	Ours wo	Ours
Bedroom	35.0524	25.6327	34.4377	**36.7279**	35.9827	35.5340	0.9730	0.9235	0.9707	**0.9774**	0.9736	0.9725
Dining room	36.3385	25.8633	**37.9529**	36.8787	37.3093	37.0157	0.9798	0.9063	0.9699	**0.9812**	0.9789	0.9757

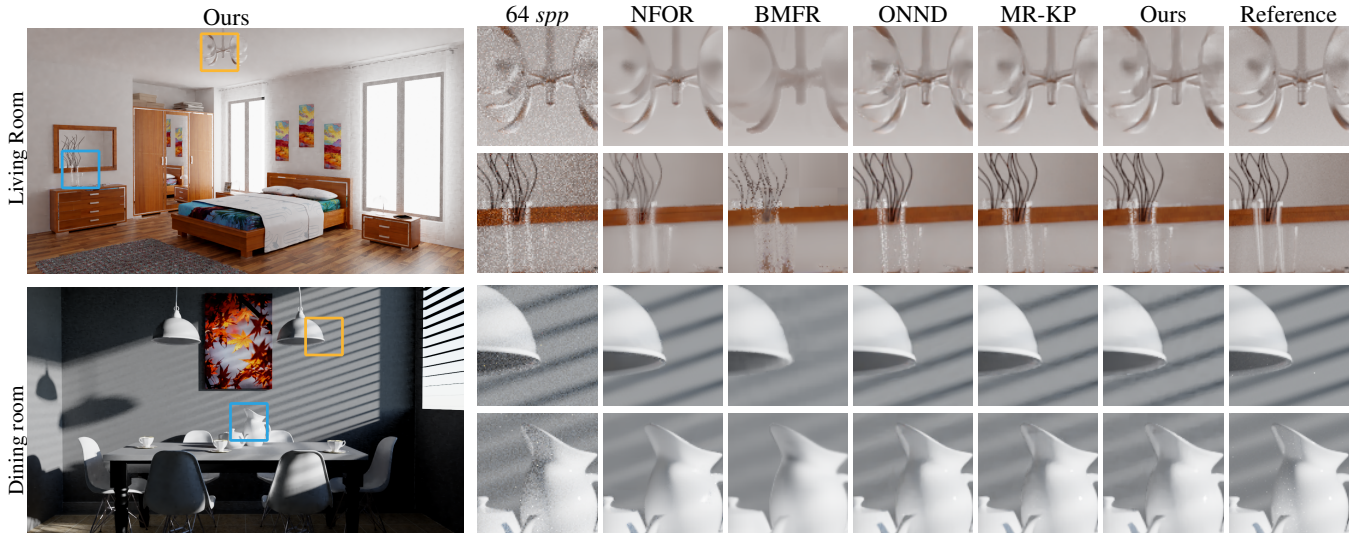

Figure 6: *Visual quality comparisons on the Tungsten dataset rendered at 64*spp*. We show a single frame from the animated sequences and do not use temporal accumulation (see additional comparisons on three other test scenes in the supplementary document). We use the 7-layer 3-grid architecture. For each scene, orange insets are shown on the top row and blue insets are on the bottom row.*

Table 5: *Average PSNR values of images from three grid scales, the arithmetic mean of the three scales, and the final output using weighted blending. The PSNR of individual scales and the mean are consistently lower than the the final output.*

Scene	Scale 0	Scale 1	Scale 2	3-scale Mean	Final output
Living room	29.7854	25.6374	32.2906	29.2378	32.2937
Sponza mov. light	23.5954	24.4703	24.7603	24.2754	24.8176
Sponza glossy	25.2993	28.8369	29.4113	27.8492	29.5481
Classroom	30.4996	29.3692	31.1660	30.3449	31.5185
San Miguel	19.0538	22.9720	23.4500	21.8253	23.6504

[CPD07] CHEN J., PARIS S., DURAND F.: Real-time edge-aware image processing with the bilateral grid. In *ACM Transactions on Graphics (TOG)* (2007), vol. 26, ACM, p. 103. 3

[DSHL10] DAMMERTZ H., SEWTZ D., HANIKA J., LENSCH H.: Edge-avoiding à-trous wavelet transform for fast global illumination filtering. In *Proceedings of the Conference on High Performance Graphics* (2010), Eurographics Association, pp. 67–75. 2

[ETH*09] EGAN K., TSENG Y.-T., HOLZSCHUCH N., DURAND F., RAMAMOORTHI R.: Frequency analysis and sheared reconstruction for rendering motion blur. In *ACM Transactions on Graphics (TOG)* (2009), vol. 28, ACM, p. 93. 2

[GCB*17] GHARBI M., CHEN J., BARRON J. T., HASINOFF S. W., DURAND F.: Deep bilateral learning for real-time image enhancement. *ACM Transactions on Graphics (TOG) 36*, 4 (2017), 118. 1, 3

[GLA*19] GHARBI M., LI T.-M., AITTALA M., LEHTINEN J., DURAND F.: Sample-based monte carlo denoising using a kernel-splatting network. *ACM Transactions on Graphics (TOG) 38*, 4 (2019), 125. 1, 2

[HLVDMW17] HUANG G., LIU Z., VAN DER MAATEN L., WEINBERGER K. Q.: Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (2017), pp. 4700–4708. 7

[HST10] HE K., SUN J., TANG X.: Guided image filtering. In *European conference on computer vision* (2010), Springer, pp. 1–14. 2

[KB14] KINGMA D. P., BA J.: Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014). 7

[KBS15] KALANTARI N. K., BAKO S., SEN P.: A machine learning approach for filtering monte carlo noise. *ACM Trans. Graph. 34*, 4 (2015), 122–1. 2, 6

[KHL19] KETTUNEN M., HÄRKÖNEN E., LEHTINEN J.: Deep convolutional reconstruction for gradient-domain rendering. *ACM Transactions on Graphics (TOG) 38*, 4 (2019), 126. 2

[KIM*19] KOSKELA M., IMMONEN K., MÄKITALO M., FOI A., VIITANEN T., JÄÄSKELÄINEN P., KULTALA H., TAKALA J.: Blockwise multi-order feature regression for real-time path-tracing reconstruction. *ACM Transactions on Graphics (TOG) 38*, 5 (2019), 138. 2, 6, 7

[KKR18] KUZNETSOV A., KALANTARI N. K., RAMAMOORTHI R.: Deep adaptive sampling for low sample count rendering. In *Computer Graphics Forum* (2018), vol. 37, Wiley Online Library, pp. 35–44. 2

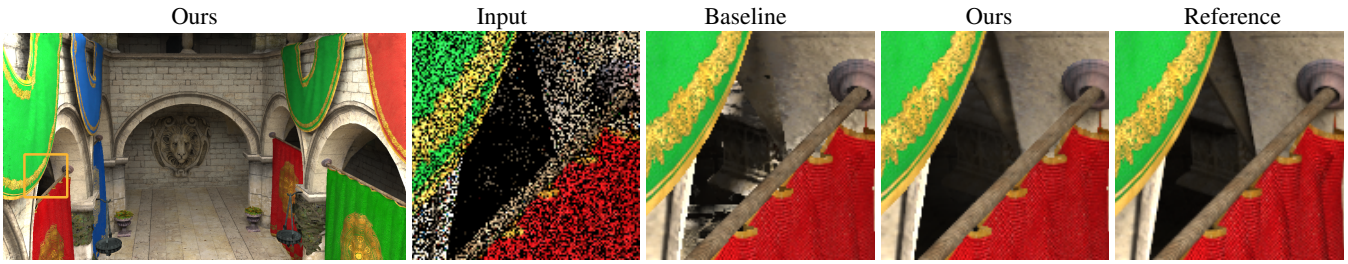

Figure 7: *Comparisons between our method and a baseline that computes guide images by simply averaging all auxiliary features. The test data is a 1-*spp *rendered image from the Sponza scene.*

Figure 8: *We visualize a baseline approach using the arithmetic mean of the auxiliary feature channels as guide image, and our three optimized guide images of the three grid scales produced by the trained GuideNet. As shown in the closeups, our guides effectively instruct the correct placement of radiance around edges.*

[KMA*15] KETTUNEN M., MANZI M., AITTALA M., LEHTINEN J., DURAND F., ZWICKER M.: Gradient-domain path tracing. *ACM Transactions on Graphics (TOG) 34*, 4 (2015), 123. 2

[KS13] KALANTARI N. K., SEN P.: Removing the noise in monte carlo rendering with general image denoising algorithms. In *Computer Graphics Forum* (2013), vol. 32, Wiley Online Library, pp. 93–102. 2

[LAC*11] LEHTINEN J., AILA T., CHEN J., LAINE S., DURAND F.: Temporal light field reconstruction for rendering distribution effects. In *ACM Transactions on Graphics (TOG)* (2011), vol. 30, ACM, p. 55. 2

[LWC12] LI T.-M., WU Y.-T., CHUANG Y.-Y.: Sure-based optimization for adaptive sampling and reconstruction. *ACM Transactions on Graphics (TOG) 31*, 6 (2012), 194. 2

[MCY14] MOON B., CARR N., YOON S.-E.: Adaptive rendering based on weighted local regression. *ACM Transactions on Graphics (TOG) 33*, 5 (2014), 170. 2

[MMBJ17] MARA M., MCGUIRE M., BITTERLI B., JAROSZ W.: An efficient denoising algorithm for global illumination. In *High Performance Graphics* (2017), pp. 3–1. 2

[NH10] NAIR V., HINTON G. E.: Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th international conference on machine learning (ICML-10)* (2010), pp. 807–814. 4

[ODR09] OVERBECK R. S., DONNER C., RAMAMOORTHI R.: Adaptive wavelet rendering. *ACM Trans. Graph. 28*, 5 (2009), 140. 2

[PD06] PARIS S., DURAND F.: A fast approximation of the bilateral filter using a signal processing approach. In *European conference on computer vision* (2006), Springer, pp. 568–580. 3

[RKZ11] ROUSSELLE F., KNAUS C., ZWICKER M.: Adaptive sampling and reconstruction using greedy error minimization. In *ACM Transactions on Graphics (TOG)* (2011), vol. 30, ACM, p. 159. 2

[RKZ12] ROUSSELLE F., KNAUS C., ZWICKER M.: Adaptive rendering with non-local means filtering. *ACM Transactions on Graphics (TOG) 31*, 6 (2012), 195. 2

[RMZ13] ROUSSELLE F., MANZI M., ZWICKER M.: Robust denoising using feature and color information. In *Computer Graphics Forum* (2013), vol. 32, Wiley Online Library, pp. 121–130. 2

[SKW*17] SCHIED C., KAPLANYAN A., WYMAN C., PATNEY A., CHAITANYA C. R. A., BURGESS J., LIU S., DACHSBACHER C., LEFOHN A., SALVI M.: Spatiotemporal variance-guided filtering: real-time reconstruction for path-traced global illumination. In *Proceedings of High Performance Graphics* (2017), ACM, p. 2. 2, 6, 7

[SPD18] SCHIED C., PETERS C., DACHSBACHER C.: Gradient estimation for real-time adaptive temporal filtering. *Proceedings of the ACM on Computer Graphics and Interactive Techniques 1*, 2 (2018), 24. 2

[TM98] TOMASI C., MANDUCHI R.: Bilateral filtering for gray and color images. In *ICCV* (1998), vol. 98, p. 2. 2, 3

[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 Transactions on Graphics (TOG) 37*, 4 (2018), 124. 1, 2, 7

[Wik18] WIKIPEDIA: Turing (microarchitecture), 2018. URL: https://en.wikipedia.org/wiki/Turing_(microarchitecture). 1

[WSB03] WANG Z., SIMONCELLI E. P., BOVIK A. C.: Multiscale structural similarity for image quality assessment. In *The Thrity-Seventh Asilomar Conference on Signals, Systems & Computers, 2003* (2003), vol. 2, Ieee, pp. 1398–1402. 7

[XZW*19] XU B., ZHANG J., WANG R., XU K., YANG Y., LI C., TANG R.: Adversarial monte carlo denoising with conditioned auxiliary feature modulation. *ACM Transactions on Graphics 38*, 6 (2019), 224. 2

[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. In *Computer Graphics Forum* (2015), vol. 34, Wiley Online Library, pp. 667–681. 2