Zeng Z, Wang L, Wang BB *et al.* Denoising stochastic progressive photon mapping renderings using a multi-residual network. JOURNAL OF COMPUTER SCIENCE AND TECHNOLOGY 35(3): 506–521 May 2020. DOI 10.1007/s11390-020-0264-1

Denoising Stochastic Progressive Photon Mapping Renderings Using a Multi-Residual Network

Zheng Zeng[1], Lu Wang[1,*], *Member*, *CCF*, *ACM*, Bei-Bei Wang[2,*], *Member*, *CCF*
Chun-Meng Kang[3], *Member*, *CCF*, *IEEE*, and Yan-Ning Xu[1], *Member*, *CCF*

[1] *School of Software, Shandong University, Jinan 250101, China*

[2] *School of Computer Science and Engineering, Nanjing University of Science and Technology, Nanjing 210094, China*

[3] *School of Information Science and Engineering, Shandong Normal University, Jinan 250358, China*

E-mail: zz.fancyvin@foxmail.com; luwang_hcivr@sdu.edu.cn; beibei.wang@njust.edu.cn; kcm89kimi@163.com
xyn@sdu.edu.cn

Received January 2, 2020; revised March 24, 2020.

Abstract Stochastic progressive photon mapping (SPPM) is one of the important global illumination methods in computer graphics. It can simulate caustics and specular-diffuse-specular lighting effects efficiently. However, as a biased method, it always suffers from both bias and variance with limited iterations, and the bias and the variance bring multi-scale noises into SPPM renderings. Recent learning-based methods have shown great advantages on denoising unbiased Monte Carlo (MC) methods, but have not been leveraged for biased ones. In this paper, we present the first learning-based method specially designed for denoising-biased SPPM renderings. Firstly, to avoid conflicting denoising constraints, the radiance of final images is decomposed into two components: caustic and global. These two components are then denoised separately via a two-network framework. In each network, we employ a novel multi-residual block with two sizes of filters, which significantly improves the model's capabilities, and makes it more suitable for multi-scale noises on both low-frequency and high-frequency areas. We also present a series of photon-related auxiliary features, to better handle noises while preserving illumination details, especially caustics. Compared with other state-of-the-art learning-based denoising methods that we apply to this problem, our method shows a higher denoising quality, which could efficiently denoise multi-scale noises while keeping sharp illuminations.

Keywords denoising, stochastic progressive photon mapping (SPPM), deep learning, residual neural network

1 Introduction

Stochastic progressive photon mapping (SPPM) [1] is a widely used global illumination simulation algorithm in computer graphics. It is a memory-friendly modification of photon mapping (PM) [2] and progressive photon mapping (PPM) [3]. It can simulate caustics and specular-diffuse-specular lighting effects efficiently.

However, as a biased but consistent method, SPPM always suffers from both bias and variance [4, 5] with limited iterations or inappropriate scenario configurations, and it often takes a prohibitive amount of time to produce a high-quality and noise-free image. The bias in SPPM renderings usually appears as low-frequency and large noises on diffuse surfaces, and the variance appears as high-frequency and small noises on glossy surfaces. This property of SPPM brings multi-scale noises into one rendering image.

Recently, deep learning approaches have shown great advantages on denoising unbiased Monte Carlo (MC) path tracing [6–9], but have not been leveraged for biased methods. In this paper, we present the first learning-based image-space denoising method designed for biased SPPM renderings (see Fig.1).

Regular Paper
Special Section of CVM 2020
This work was partially supported by the National Key Research and Development Program of China under Grant No. 2017YFB0203000, the National Natural Science Foundation of China under Grant Nos. 61802187, 61872223, and 61702311, and the Natural Science Foundation of Jiangsu Province of China under Grant No. BK20170857.

*Corresponding Author

Fig.1. We introduce a deep learning approach for denoising bias and variance in SPPM renderings. We present a network that extracts spatial features via multi-residual blocks, predicts kernel weights to filter noisy colors, and produces a high-quality result. The network is trained to learn the complex relationship between noisy inputs and corresponding reference. And then it can be used to denoise images from other scenes and output high-quality results. We show the denoising results of our method in the above images.

To avoid conflicting denoising constraints and keep more sharp caustics, we first separate radiance of final results into two components: caustic and global, which represent caustics and other illumination effects respectively. Then we employ a two-network framework to denoise these components separately, and combine them to get the final result at the end.

When denoising SPPM, a network with a larger filter size can better remove multi-scale noises, especially large noises on a low-frequency area (e.g., a flat surface). But it is difficult to remove noises without over-smoothing on a high-frequency area (e.g., a narrow region with dramatic changes in geometry). A network with a small filter size is the opposite. Based on this observation, we introduce a novel multi-residual block within each network. The block contains two residual functions with different filter sizes. It significantly improves the model's capabilities and benefits from both large and small sizes of filters when denoising. We also present some photon-related auxiliary features, such as photon density, photon flux, and photon gradient, to better handle noises while preserving correct illumination details, especially caustics. Our contributions are summarized as follows:

• the first learning-based method for biased SPPM denoising, which outperforms the state-of-the-art learning-based MC denoising methods that we apply to this problem;

• a novel deep residual denoising network with multi-residual blocks that benefits from both large and small sizes of filters, which allows better dealing with multi-scale noises on both low-frequency and high-frequency areas in SPPM renderings.

• a series of photon-related auxiliary features to better handle noises while preserving correct illumination details, especially caustics.

The remainder of this paper is organized as follows. In Section 2, we briefly review learning-based denoising approaches and photon mapping algorithms. The theoretical background and our framework for SPPM denoising are described in Section 3 and Section 4 respectively. We present our experimental setups in Section 5 and our results in Section 6. And finally, we conclude the paper in Section 7.

2 Related Work

2.1 Deep Learning Based Monte Carlo Denoising

Deep learning has shown impressive impacts on unbiased Monte Carlo (MC) denoising. The work of [10] by Kalantari *et al.* is the first using a neural network for MC denoising. They learned the relationship between the noisy scene data and the ideal filter parameters with a multilayer perceptron neural network and used the learned model for new scenes for a wide range of distributed effects. Bako *et al.*[6] introduced a novel, kernel-prediction network (KPCN) which uses the convolutional neural network (CNN) to estimate the local weighting kernels to compute each denoised pixel from its neighbors. They decomposed the diffuse and specular components, pre-trained two networks for them separately, and then fine-tuned the complete framework. KPCN has shown great improvement over the prior MC denoisers. This work was further improved by Vogels *et al.*[7], combining KPCN with several task-specific modules, and optimizing the assembly using an asymmetric loss, resulting in a more robust solution in the end. Especially, they built upon a multi-scale architecture to address residual low-frequency noise, while their single-frame denoiser could not directly address this issue. Wong and Wong[8,9] used residual learning for high-quality denoising of path tracing rendering.

Their model (RDP) directly predicts the corresponding noise-free color instead of per-pixel kernel weights. Xu *et al.*[11] also denoised MC renderings via directly predicting the final color with a generative adversarial network (GAN). It is the first adversarial learning-based method for MC denoising. And they also adapted a well-designed conditioned auxiliary feature modulation method to better utilize the rendering feature buffers. Likewise, Yang *et al.*[12] introduced a dual-encoder network (DEMC) with a feature fusion sub-network to handle the feature buffers separately. They first fused the feature buffers and then encoded these buffers with a noisy image simultaneously, and finally produced a noise-free image. Gharbi *et al.*[13] presented the first CNN that can learn to denoise MC renderings directly from the samples, rather than the summary statistics of a pixel's sample distribution. But we do not consider their method as a solution to SPPM denosing, since SPPM is a pixel-based method, rather than sample-based.

Different from previous work, we use deep learning on biased SPPM denoising. It suffers from both bias and variance, unlike the unbiased denoising method that only focuses on the variance issue.

2.2 Photon Mapping

Photon mapping (PM)[2] is one of the important global illumination methods. It is robust and effective at simulating caustics and specular-diffuse-specular (SDS) light paths. These are challenging for unbiased general MC based methods. PM runs in two steps: photon tracing and photon density estimation. In the first step, the photons are shot from the light sources, bounced in the scene and stored in photon maps. In the second step, the radiance of each pixel is computed by estimating the photon density in the photon maps. However, PM suffers from memory issue, as it often requires storing an infinite number of photons to produce a photometrically correct result[14].

Progressive photon mapping (PPM) methods[3] address the memory issue by restructuring PM to iteratively trace new photons. It was further modified by Hachisuka *et al.*[1] to stochastic progressive photon mapping (SPPM), which generates new gather points in each iteration for circumventing the memory bound. However, SPPM still suffers from both bias and variance[4,5] with limited iterations or inappropriate scenario configurations.

Several attempts have been made to reduce bias and variance for PM and PPM, by changing photon energy accumulation[2,15] or improving the searching bandwidth[16]. Photon relaxation methods[17] eliminate random distribution noise in the photon map. Fu *et al.*[18] adjusted a weighting parameter adaptively across the scene for a better balance between bias and variance. Kaplanyan *et al.*[19] introduced adaptive progressive photon mapping (APPM), which optimally balances bias and variance to minimize the overall error.

All these approaches highly rely on the 3D space with great complexity. Compared with them, we resort to the image-space denoising solution which is simple and effective.

2.3 Deep Residual Networks

Increasing the network depth is known to improve the model capabilities. However, increasing the depth can be challenging for the learning process because of the vanishing gradient problem[20]. Deep residual networks[21] address this issue by using identity skip-connections. These skip-connections and residual blocks in the network permit the reuse of upstream features to establish a multi-scale alike mapping capability and maintain some consistency with upstream blocks[8].

Specifically, for denoising task, it always needs to be designed relatively deep to improve the model capabilities while maintaining some consistency with the input. This makes the deep residual network naturally suited for the denoising task[8].

However, an obvious drawback of residual networks is that every percentage of improvement requires significantly increasing the number of layers[22], which linearly increases the input image size when predicting the denoiser kernel[7].

Recently, some theories have been put forward, that the width (the number of convolutional filters)[23] and the multiplicity (the number of residual functions in the residual blocks)[24] are important to improve the model capabilities in addition to depth. And inspired by Abdi *et al.*[24], we propose to increase the multiplicity of SPPM denoising network, i.e., to employ a multi-residual block, which can significantly improve the training performance and better handle the multi-scale noises while keeping the depth fixed.

3 Theoretical Background

In this section, we first introduce the bias and variance issues of SPPM. Then we formalize the SPPM

denoising as a supervised learning problem and define some necessary notations.

3.1 Bias and Variance in SPPM

SPPM [1] is a multi-pass method. In the distributed ray tracing pass, a set of hit points is generated randomly. In the photon tracing pass, a number of photons are shot and traced from lights, accumulating contributions at nearby hit points, and then assigning shared statistics (e.g., shared accumulated flux, shared searching radius) to each pixel. This process then repeats.

Using the shared statistics, the average radiance value $L(S, \boldsymbol{\omega})$ at the i-th pass over the region S toward the direction $\boldsymbol{\omega}$ is approximated:

$$L(S, \boldsymbol{\omega}) \approx \frac{\tau_i(S, \boldsymbol{\omega})}{N_{\mathrm{e}}(i)\pi R_i(S)^2},$$

where i is the photon tracing iteration, $\tau_i(S, \boldsymbol{\omega})$ is the shared accumulated flux over the region S, $N_e(i)$ is the number of emitted photons after i passes, and $R_i(S)$ is the shared searching radius.

These shared statistics are then updated:

$$N_{i+1}(S) = N_i(S) + \alpha M_i(\boldsymbol{x}_i), \tag{1}$$

$$R_{i+1}(S) = R_i(S)\sqrt{\frac{N_i(S) + \alpha M_i(\boldsymbol{x}_i)}{N_i(S) + M_i(\boldsymbol{x}_i)}},$$

$$\Phi_i(\boldsymbol{x}_i, \boldsymbol{\omega}) = \sum_{p=1}^{M_i(\boldsymbol{x}_i)} f_r(\boldsymbol{x}_i, \boldsymbol{\omega}, \boldsymbol{\omega}_p)\, \Phi_p(\boldsymbol{x}_p, \boldsymbol{\omega}_p), \tag{2}$$

$$\tau_{i+1}(S, \boldsymbol{\omega}) = (\tau_i(S, \boldsymbol{\omega}) + \Phi_i(\boldsymbol{x}_i, \boldsymbol{\omega}))\frac{R_{i+1}(S)^2}{R_i(S)^2},$$

where $\boldsymbol{x}_i$ is a randomly generated position within S, $N_i(S)$ is the shared local photon count, M_i is the found photon count within the searching radius during the i-th pass, f_r is the bidirectional reflectance distribution function (BRDF) and $\Phi_p(\boldsymbol{x}_p, \boldsymbol{\omega}_p)$ is the flux of photon p with position $\boldsymbol{x}_i$ and incoming direction $\boldsymbol{\omega}_p$. $\alpha \in (0, 1)$ is a user-defined parameter that determines how quickly the contributions from photons in earlier passes are faded out.

Since this updating procedure satisfies the conditions of consistency [25], the stochastic radiance estimate converges to the correct average radiance over S for $i \to \infty$:

$$L(S, \boldsymbol{\omega}) = \lim_{i \to \infty} \frac{\tau_i(S, \boldsymbol{\omega})}{N_{\mathrm{e}}(i)\pi R_i(S)^2},$$

meanwhile $N_{\mathrm{e}}(i) \to \infty$ and $R_i(S) \to 0$.

However, in practice, this radiance estimate is hard to converge to the correct result. And SPPM always suffers from both bias and variance under a finite number of passes i, or inappropriate user-supplied scenario configurations (e.g., α, $R_{\mathrm{inital}}(S)$, or the number of emitted photons per pass).

As shown in Fig.2, bias usually appears on diffuse surfaces as low-frequency and relatively large noises, or over blurring illumination features (e.g., soft shadows and caustics). This is due to the biased radiance estimation at hit points with an insufficient number of photons $N_{\mathrm{e}}(i)$ or an overlarge searching radius $R_i(S)$.

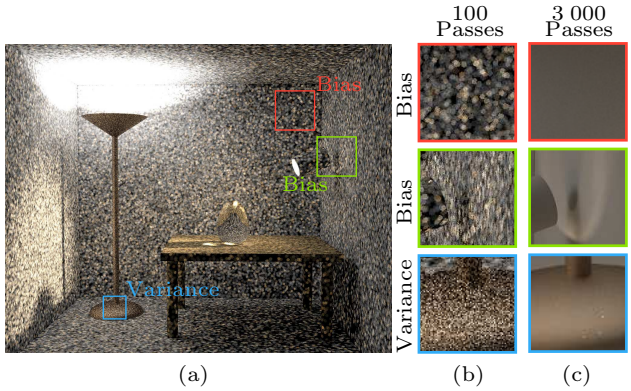

Fig. 2. Bias and variance in SPPM renderings. Bias usually appears as low-frequency and large noises, and variance usually appears as high-frequency and small noises. (a) SPPM rendering result on Bidir Room scene, 100 rendering passes. (b) A zoom-in view of 100 passes result. (c) A zoom-in view of 3 000 passes result.

On glossy surfaces (see Fig.2), the variance looks like high-frequency and relatively small noise. This is because glossy surfaces may have high variance if the specular lobe is tight and not enough photons have arrived to represent the incident radiance distribution well [26]. Furthermore, sampling from a glossy surface to get the next hit point also leads to variance.

3.2 Image-Space Denoising Model for SPPM

The bias and the variance bring multi-scale noises into one rendering of SPPM. For solving it, we propose an image-space denoising method by modeling and estimating a filtering function G. It is used to compute the filtered color $\hat{\boldsymbol{c}}_i$ of a pixel i. $\hat{\boldsymbol{c}}_i$ is evaluated as the weighted sum of the pixel colors $\boldsymbol{c}_j$ in a neighborhood $\mathcal{N}(i)$ centered at pixel i:

$$\hat{\boldsymbol{c}}_i = \sum_{j \in \mathcal{N}(i)} G(X_i, \theta_{i,j})\, \boldsymbol{c}_j, \tag{3}$$

where $X_i = \{\boldsymbol{x}_j : j \in \mathcal{N}(i)\}$ is a block of vectors around the neighborhood $\mathcal{N}(i)$, and $\boldsymbol{x}_j = (\boldsymbol{c}_j, \boldsymbol{f}_j)$ is a per-pixel

vector including color $\boldsymbol{c}_j$ and auxiliary features $\boldsymbol{f}_j$ computed from the photon map. $\theta_{i,j}$ represents the weight of color $\boldsymbol{c}_j$.

Bako *et al.* [6] modelled a similar function of G via a deep fully convolutional network (KPCN) to predict the per-pixel kernel. KPCN achieves state-of-the-art performance at removing high-frequency and small noise in path tracing (PT) renderings, but it tends to produce low-frequency artifacts [7], which makes it even more un-robust to handle low-frequency and large noises in SPPM renderings.

Especially, when denoising multi-scale noises in SPPM renderings, it needs a large filter size for better removing large noises on low-frequency areas. But filters with large size always fail to remove noises without over-smoothing on high-frequency areas, like a corner of wall or cabinet seams. And filters with a small size are just on the contrary.

To address this limitation, we propose a learning-based method with a novel multi-residual denoising network to model function G for SPPM, which benefits from both large and small sizes of filters.

4 Deep Residual Denoising

In this section, we introduce an SPPM denoising framework to model the function G in (3) with a multi-residual denoising network (MRDN) (Fig.3). We first separate the radiance of final renderings into caustic and global components (Subsection 4.1). Then, we use a two-network framework (Subsection 4.2) to denoise them. For each component, we employ an MRDN to deal with multi-scale noises from bias and variance. We also use a set of photon specific auxiliary features (Subsection 4.3) as the network inputs to better handle bias while preserving sharp illuminations.

4.1 Components Decomposition

Using a weighted reconstruction method (like the function G to directly denoise SPPM) always leads to conflicting denoising constraints. Suppose a group of photons contribute to a surface, it is hard to decide whether to average out and blur out their contributions (they are a biased estimate on the diffuse surface) or to stay sharp (they are caustics with some fuzzy).

We address this issue by decomposing the rendering result, i.e., final radiance of each pixel, into two components: caustic and global. These two components represent the radiance estimation result of two types of photons: caustic photons and global photons. If photons go through a delta BSDF and reach a diffuse surface or reach a glossy surface at the maximum depth without any diffuse bounces, they are called caustic photons; otherwise, global photons.

The two components are pre-processed, denoised, and post-processed respectively, and then get combined to produce the final result (Fig.3). Thus, we reform (3) as:

$$\hat{\boldsymbol{c}}_{\text{global}_i} = \sum_{j\in\mathcal{N}(i)} G_{\text{global}}\left(\boldsymbol{X}_{\text{global}_i}, \theta_{i,j}\right)\boldsymbol{c}_{\text{global}_j},$$

$$\hat{\boldsymbol{c}}_{\text{caustic}_i} = \sum_{j\in\mathcal{N}(i)} G_{\text{caustic}}\left(\boldsymbol{X}_{\text{caustic}_i}, \theta_{i,j}\right)\boldsymbol{c}_{\text{caustic}_j},$$

$$\hat{\boldsymbol{c}}_i = \hat{\boldsymbol{c}}_{\text{global}_i} + \hat{\boldsymbol{c}}_{\text{caustic}_i}.$$

Both caustic and global components have high-dynamic range (HDR) values, which make the optimization process highly unstable; thus we apply a logarithmic transform in the pre-processing step to each component:

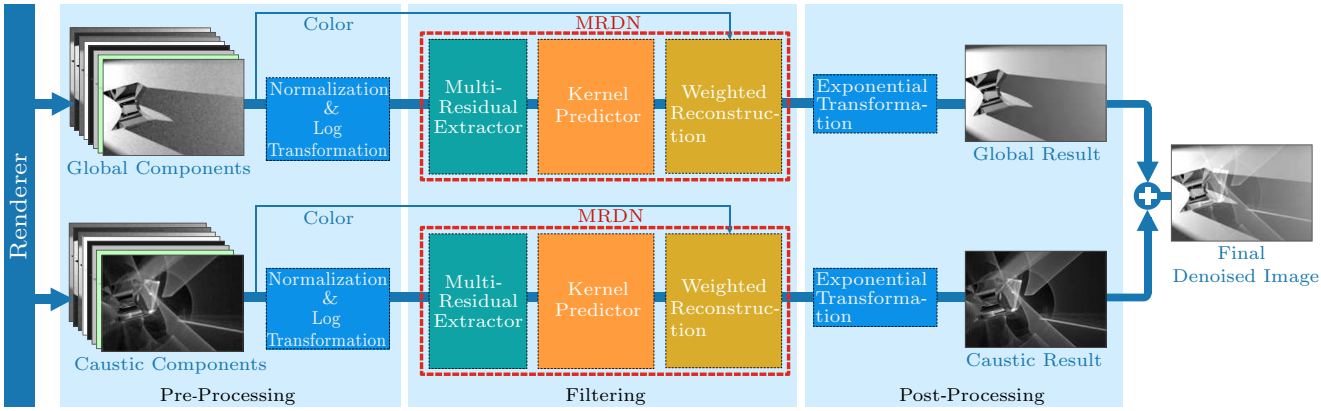

Fig. 3. Overview of our denoising framework. We start by preprocessing caustic and global components and then feed them to the two networks which denoise the two components respectively. After denoising, the two separate outputs are reconstructed and post-processed to get the final denoised image.

$\widetilde{\boldsymbol{c}} = \log(1+\boldsymbol{c})$. After being denoised separately, we apply exponential transformation to each network outputs and then recombine them:

$$\hat{\boldsymbol{c}} = \exp(\hat{\boldsymbol{c}}_{\text{global}}) + \exp(\hat{\boldsymbol{c}}_{\text{caustic}}) - 2.$$

4.2 Network Architecture

We propose a multi-residual denoising network (MRDN) (Fig.4). It consists of a novel multi-residual extractor and a kernel predictor with weighted reconstruction designed by Vogels *et al.*[7]

The multi-residual extractor extracts the spatial features, which has nine multi-residual blocks along with convolution, activation, and batch-normalize layers. We use two residual functions with different receptive fields (i.e., filter kernel sizes) in each residual block (Fig.5), to better handle the multi-scale noises on both low-frequency and high-frequency areas:

$$H(\boldsymbol{x}) = Concatenate(F_1(\boldsymbol{x}), F_2(\boldsymbol{x})) + \boldsymbol{x},$$

where $H(\boldsymbol{x})$ is the output of multi-residual blocks, $F_1(\boldsymbol{x})$ and $F_2(\boldsymbol{x})$ are the different residual functions, and $\boldsymbol{x}$ is the input of blocks (skip connection). The skip connection aims to construct identity mapping and permit features reuse.

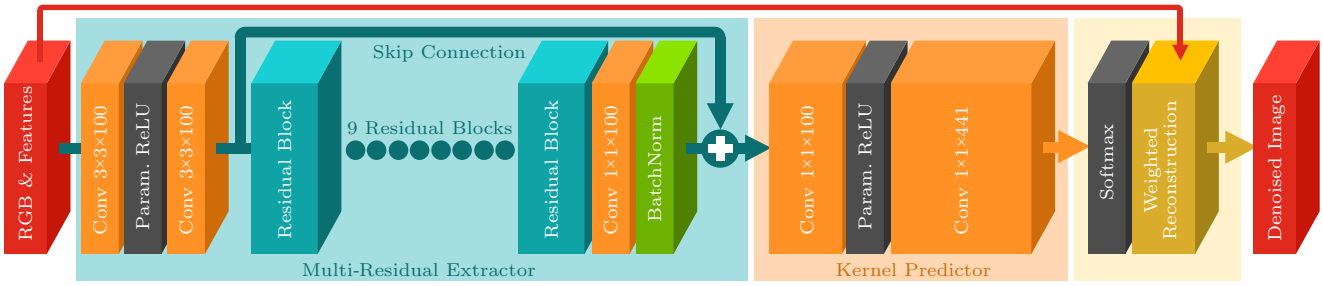

Fig.4. Overview of our multi-residual denoising network (MRDN). It consists of a novel multi-residual extractor, a kernel predictor, and weighted reconstruction. "Conv" refers to the convolutional layer.

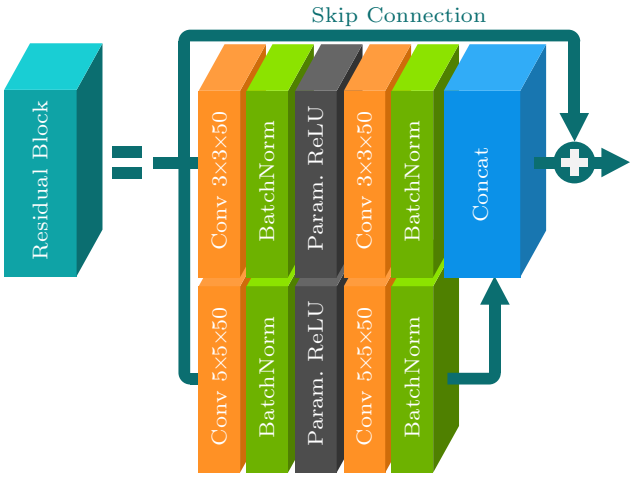

Fig. 5. Overview of our multi-residual block. There are two streams with two sizes of convolutional filters, 3×3 (up) and 5×5 (down).

After extracting the spatial features, we employ a kernel predictor, to predict 21×21 per-pixel kernels weight, and then reconstruct the final color result.

In our network, we use parametric rectified linear unit (PReLU)[27] as our activation function, as it performs the best in terms of both training efficiency and denoising quality. We also employ a set of batch normalization layers which can make our network much more robust and less sensitive to hyper-parameters.

4.3 Auxiliary Features

Besides the features from previous work[6] (e.g., normal, tracing depth, albedo), we present some other auxiliary features (as shown in Fig.6) to better handle noises while preserving illumination details, especially caustics:

- *Distance*: the distance t from the surface intersection to the camera;
- *Photon Density*: the local gathered photon count $N_i(S)$ from the per-pixel shared statistics in (1) with each pixel as one unit of area;
- *Photon Flux*: the shared accumulated flux $\tau_i(S,\omega)$ over the per-pixel in (2);
- *Photon Gradient*: the energy-direction distribution of the photons over a region S, described in previous work[28].

5 Experimental Setup

In this section, we present our dataset and describe our network implementation and training details.

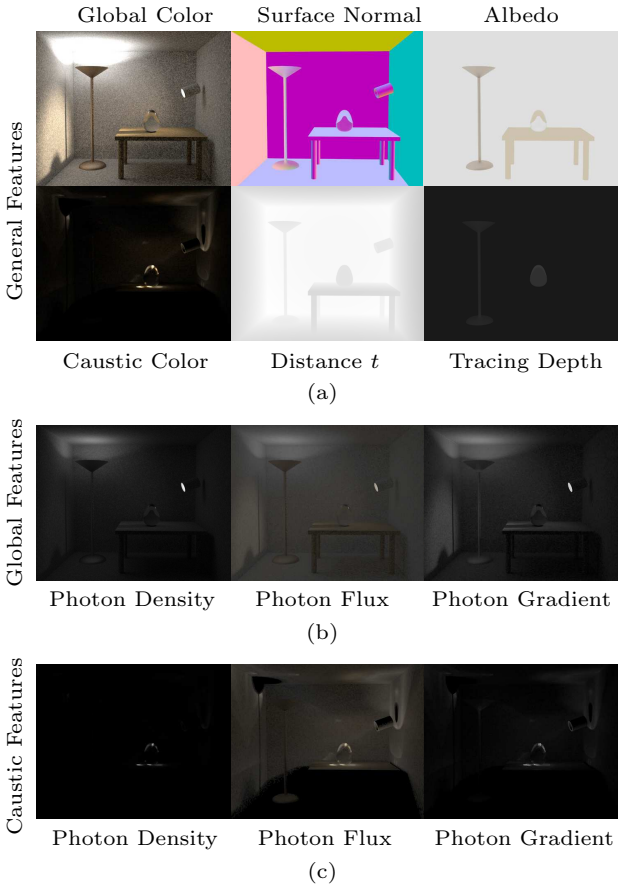

Fig. 6. Color buffers and auxiliary feature buffers. (a) General feature buffers, including the color buffer. (b) Photon-related feature buffers of global component. (c) Photon-related feature buffers of caustic component.

5.1 Dataset

A major difficulty of training a deep denoising network is that it always requires a sufficiently large and representative dataset to learn the complex relationship between its noisy inputs and high-quality outputs while avoiding overfitting.

As there is no public dataset for our network, we prepared a reasonably large high-quality dataset by taking 19 scenes curated by Bitterli *et al.*[①], covering simple scenes to complex scenes with indirect lighting. We modified the camera, light sources, and materials to generate 827 different training scenes. We held out 10% of this dataset as validation data, for preventing over-fitting and tuning the hyperparameters. We also prepared several challenging scenes with complex illumination effects as the test dataset to evaluate the final model performance.

We rendered noisy images, auxiliary features, and corresponding noise-free references with SPPM of Mitsuba renderer[②]. The reference images were rendered with 3 000 passes and 5 million photons emitted per pass. Although there still left a small number of visible noises (see Fig.7), they were converged enough for our network. For noisy images, previous work [6, 8] tends to use the same render settings for all scenes (e.g., 128 samples per pixel) since path tracing will have a similar noise pattern. As to SPPM, in one rendering image, there are multi-scale noises; besides, it always produces multi-scale noises under the same settings for different scenes. Therefore, to better improve the denoising ability of our model, we chose to use several different sets of render settings for all scenes (e.g., 100 passes and 500k photons, 10 passes and 100k photons).

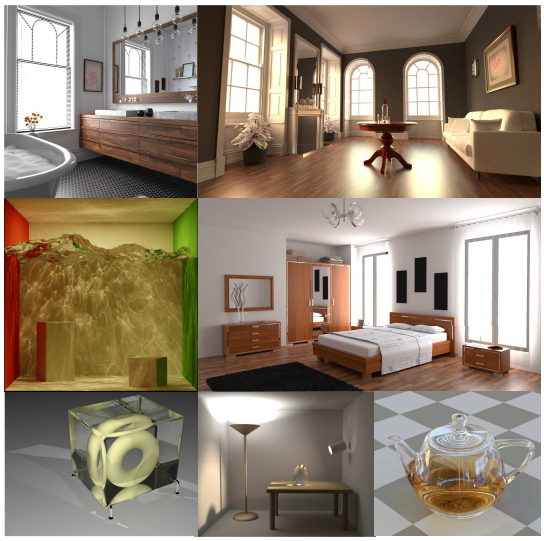

Fig. 7. Selected training images from our dataset.

And all the output auxiliary features include:

• global and caustic RGB color buffers $\boldsymbol{c}_{\text{global}}$ and $\boldsymbol{c}_{\text{caustic}}$ respectively;

• per-pixel general features buffers $\boldsymbol{f}_{\text{general}}$, consisting of normals (3 channels), albedo (3 channels), depth (1 channel), and t (1 channel);

• photon features buffers $\boldsymbol{f}_{\text{global}}$ and $\boldsymbol{f}_{\text{caustic}}$, consisting of photon density (1 channel), photon flux (3 channels), and photon gradient (3 channels).

5.2 Preprocessing

First, we pre-processed the color buffers as described in Subsection 4.1 to get $\widetilde{\boldsymbol{c}}_{\text{global}}$ and $\widetilde{\boldsymbol{c}}_{\text{caustic}}$.

①Bitterli B. Rendering resources, 2016. https://benedikt-bitterli.me/resources/, March 2020.

②Jakob W. Mitsuba renderer, 2010. http://www.mitsuba-renderer.org, March 2020.

Then, for t in $\boldsymbol{f}_{\text{general}}$ and other non-negative and high-dynamic range features (e.g., photon density, photon flux, photon gradient) in $\boldsymbol{f}_{\text{global}}$ and $\boldsymbol{f}_{\text{caustic}}$, we scaled their values via min-max normalization to the range $[0, 1]$.

Finally, we calculated the image-space gradients of colors and all features in both x and y direction for better identifying and understanding edge information and sharp illumination features.

After this processing, we constructed our network inputs $\boldsymbol{x}$ as:

$$\begin{aligned}
\boldsymbol{v}_{\text{global}} &= \{\widetilde{\boldsymbol{c}}_{\text{global}}, \boldsymbol{f}_{\text{general}}, \boldsymbol{f}_{\text{global}}\},\\
\boldsymbol{v}_{\text{caustic}} &= \{\widetilde{\boldsymbol{c}}_{\text{caustic}}, \boldsymbol{f}_{\text{general}}, \boldsymbol{f}_{\text{caustic}}\},\\
\boldsymbol{x}_{\text{global}} &= \{\boldsymbol{v}_{\text{global}}, G_x(\boldsymbol{v}_{\text{global}}), G_y(\boldsymbol{v}_{\text{global}})\},\\
\boldsymbol{x}_{\text{caustic}} &= \{\boldsymbol{v}_{\text{caustic}}, G_x(\boldsymbol{v}_{\text{caustic}}), G_y(\boldsymbol{v}_{\text{caustic}})\}.
\end{aligned}$$

From each feature image, we extracted over 400 unique patches with size 65×65 from each frame according to its resolution.

In the end, we prepared and sorted out two groups of training data as follows:

- $\boldsymbol{x}_{\text{global}}$ and its corresponding references $\bar{\boldsymbol{c}}_{\text{global}}$;
- $\boldsymbol{x}_{\text{causitc}}$ and its corresponding references $\bar{\boldsymbol{c}}_{\text{caustic}}$.

5.3 Model Implementation and Training

We implemented our networks in TensorFlow [29], optimizing them with L1-loss using ADAM [30] with a learning rate of 10^{-4}, and setting the gradient clip threshold as 1.0.

The networks of caustic and global components are trained respectively with a batch size of 128. As Bako *et al.* [6] suggested, for speeding up both training and inference, we kept the number of parameters reasonably low. Each network only has a total of 2 579 841 trainable parameters. And we initialized them with Xavier initialization [31].

During training, we evaluated the performance of our networks on a validation dataset after each epoch, to avoid over-fitting via early stopping. These two networks are trained for approximately 25 epochs to get their best performance.

6 Results

We compare our method with KPCN [6] and RDP [8]. We apply all these methods on top of our components decomposition and give the same inputs. For ensuring fairness of comparison, we list the number of trainable parameters (#parameters) and floating point operations per second (FLOPS) of these networks in Table 1.

Table 1. Trainable Parameters and FLOPS

Method	# Parameters	FLOPS
MRDN (ours)	2 579 841	5 153 429
KPCN	2 973 741	5 945 023
RDP	2 819 075	5 632 443

Note: #: Number of.

We measure image quality with mean square error (MSE) and structural similarity (SSIM). For a detailed comparison of HDR images, in Subsection 6.3, we use image difference. The image difference of two images is defined as the sum of the absolute difference at each pixel.

6.1 Denoising Quality

Fig.8 compares our network (MRDN) against KPCN and RDP on the Kitchen scene. In Figs.8–10, "x mil" and "y passes" refer to rendering with y passes and x million photons per pass. In these figures, we highlight the best scores in bold. Qualitatively, our approach produces a higher quality than the other methods. Both KPCN and RDP tend to leave low-frequency noises (e.g., on diffuse cupboards, walls, and ceilings), while our method can remove them much better. And they both lose some texture details after removing noises, while our method can preserve the details well (see the chopping board, radio, and cabinet on the table). Furthermore, RDP produces color offset artifacts (in this case, redder than the original) after denoising, due to direct color prediction rather than kernel prediction and reconstruction, which makes RDP unstable to denoise unfamiliar scenes.

In Fig.9, we compare these methods with more test scenes. The lower error measurements confirm that our method produces the higher quality. With our method, the sharp illumination details are better preserved, and both the high-frequency and low-frequency noises are better removed. In Fig.10 we compare these methods under different render settings on the same scene. Compared with others, our method achieves a higher denoising quality, and it tends to keep more illumination details despite a low photon density.

6.2 Performance Analysis

In terms of timing, for an image of $1\,920 \times 1\,080$ pixels, KPCN takes an average of 10 s to evaluate kernels and reconstruct a denoised image, while our network takes about 14 s. The reason is that our multi-residual

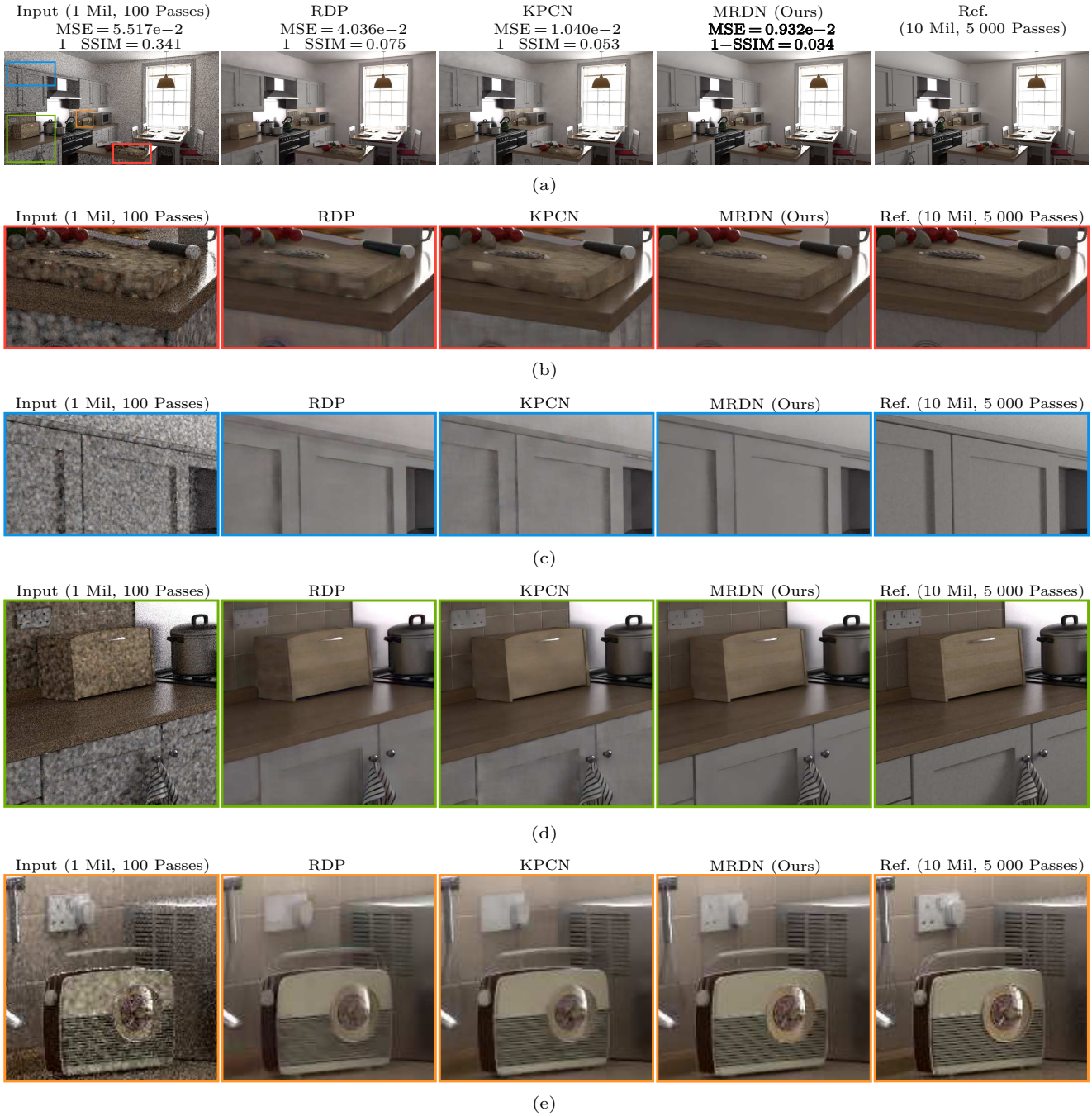

Fig.8. Comparison of denoised results from KPCN, RDP, and MRDN (ours) on a challenge scene (the Kitchen scene). The quality measurement metrics (MSE and SSIM) are shown above the images. MRDN (ours) holds the smaller errors, and it always removes more multi-scale noises while better preserving texture details. (a) Comparison of the whole image. (b) Close-up comparison of the chopping board. (c) Close-up comparison of the cupboards. (d) Close-up comparison of the cabinet on the table. (e) Close-up comparison of the radio.

block requires more time to do inference. And RDP generally takes about 15 s under the same settings.

In Fig.11, we compare the validation L1-loss of MRDN, KPCN, and RDP. Our method (MRDN) achieves better convergence performance compared with KPCN and RDP on the validation set. We observe that the convergence of RDP is extremely unstable, and it holds a much larger error. Both MRDN and KPCN have a smaller error at the early stage, due to kernel prediction and weighted reconstruction. With these two modules, even the untrained network can produce images that resemble the input, just with blurring.

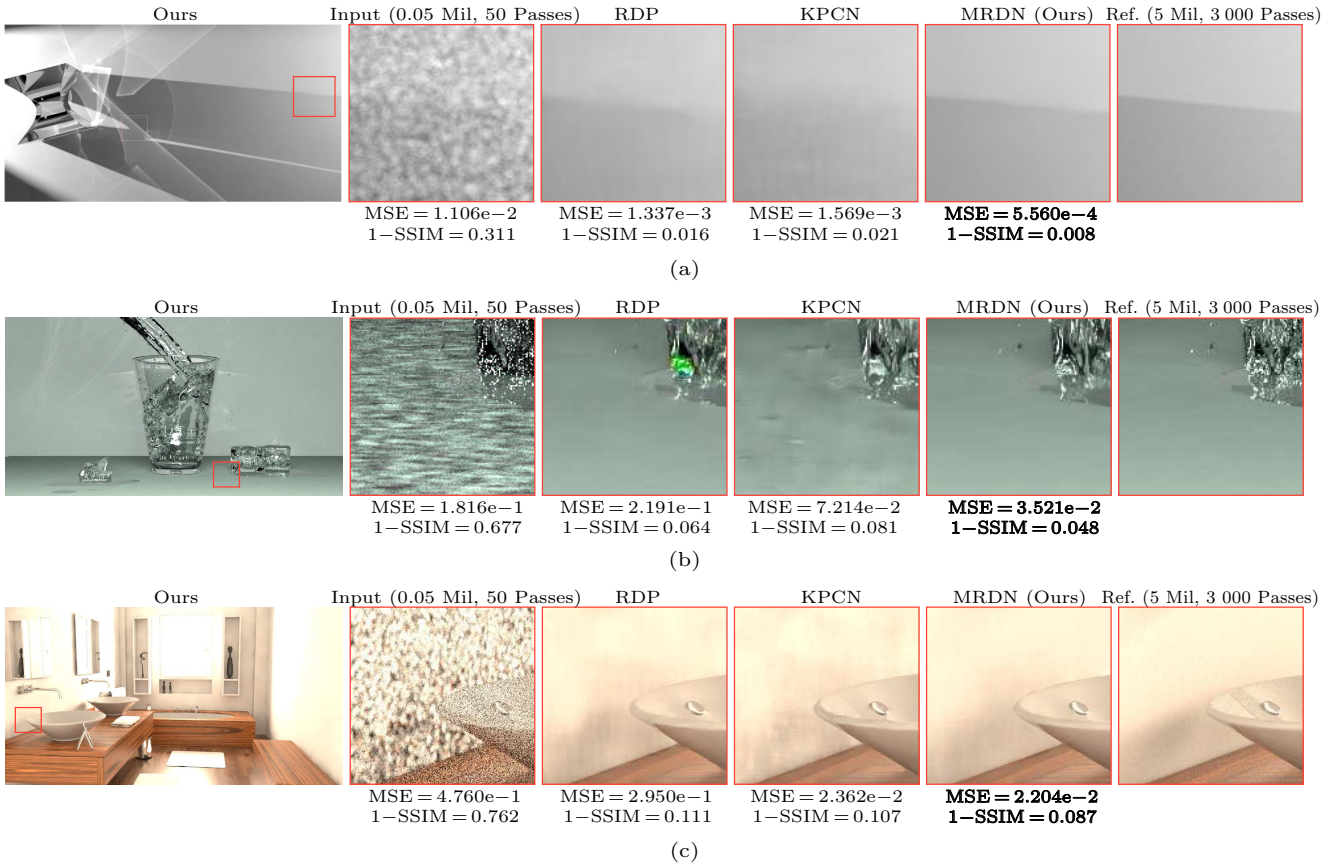

Fig. 9. Comparison of denoised results from KPCN, RDP, and MRDN (ours) on different test scenes, under a render setting of 0.05 millions photons and 50 passes. Our method removes more multi-scale noises while better preserving illumination details. (a) Comparison on Lux scene. (b) Comparison on Glass of Water scene. (c) Comparison on Salle de Bain scene.

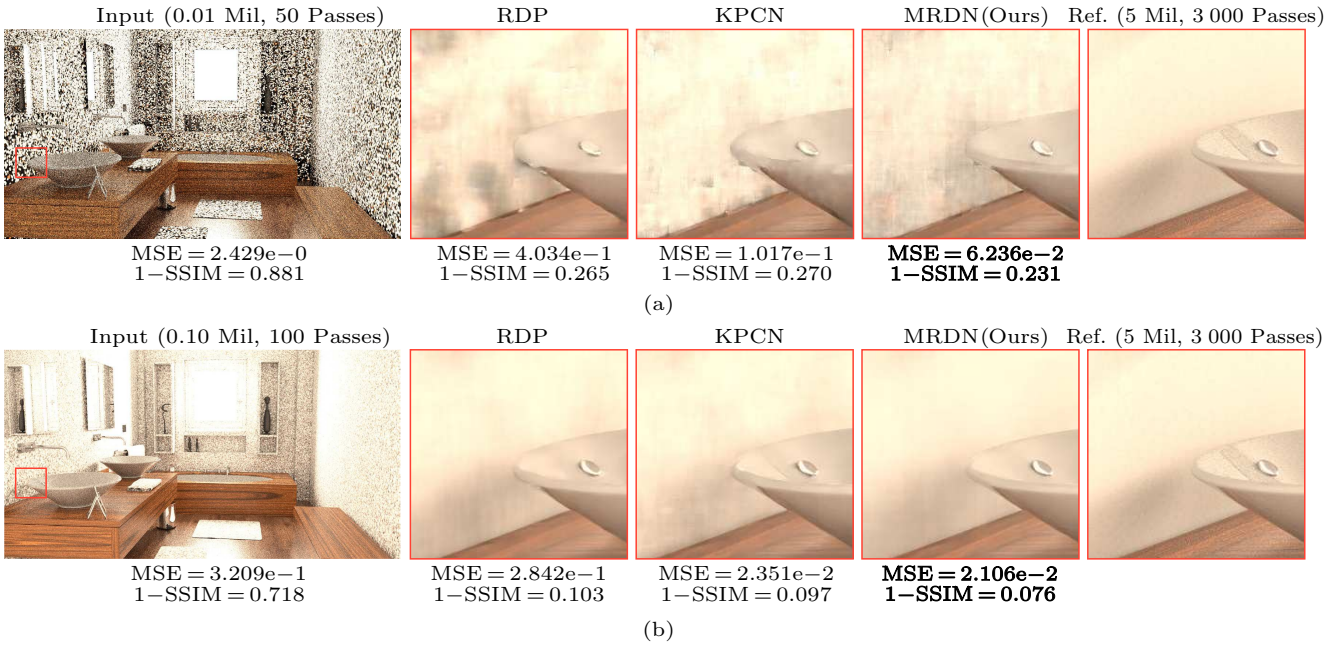

Fig.10. Comparison of denoised results from KPCN, RDP, and MRDN (ours) under different render settings. Our method achieves a higher denoising quality, and it tends to keep more illumination details. (a) Comparison on Salle de Bain scene, rendering with 0.01 million photons per pass, 50 passes. (b) Comparison on Salle de Bain scene, rendering with 0.10 million photons per pass, 100 passes.

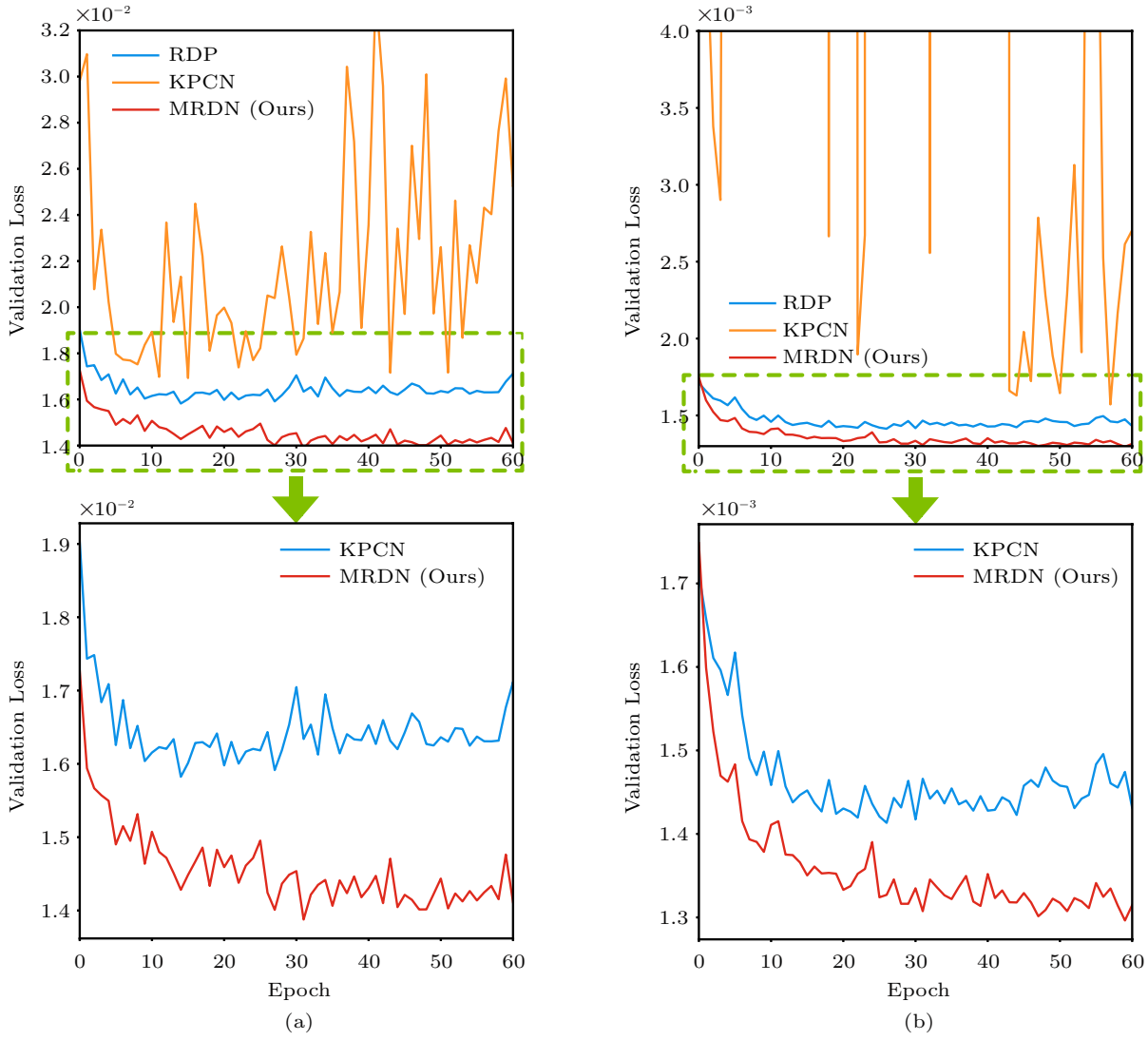

Fig.11. Convergence plots of MRDN (ours), KPCN, and RDP on (a) global and (b) caustic components. These networks are evaluated with L1-loss on our validation dataset. The upper row shows that RDP is hard to converge and has the largest error. We zoom in and show a more detailed comparison of the remaining two networks of KPCN and ours in the lower row.

To fully test the potential performance, we further increase the trainable parameters by widening the networks. As illustrated in Fig.12, MRDN gains a huge performance boost, while KPCN becomes unstable and easy to over-fit.

6.3 MRDN Analysis

MRDN vs SRDN. To further validate the effectiveness of our MRDN, we implement two different single-residual denoising networks (SRDNs) with only one residual function and one size of kernels in each residual block. SRDN-5 has larger 5×5 kernels and SRDN-3 has smaller 3×3 ones. To ensure fairness, we slightly increase the number of channels per layer in SRDNs blocks, to get a similar number of parameters to MRDN.

Fig.13 compares the validation L1-loss between the MRDN and SRDNs for both the global and the caustic components. Since the multi-scale noises issue is more serious in the global component, MRDN gains much better performance than SRDNs when denoising the global component. And in the caustic component, it is only a little better.

Fig.14 shows the image difference of denoising results and the references. We can see that compared with MRDN and SRDN-5, SRDN-3 has failed to remove low-frequency and large noises on flat cabinet sur-

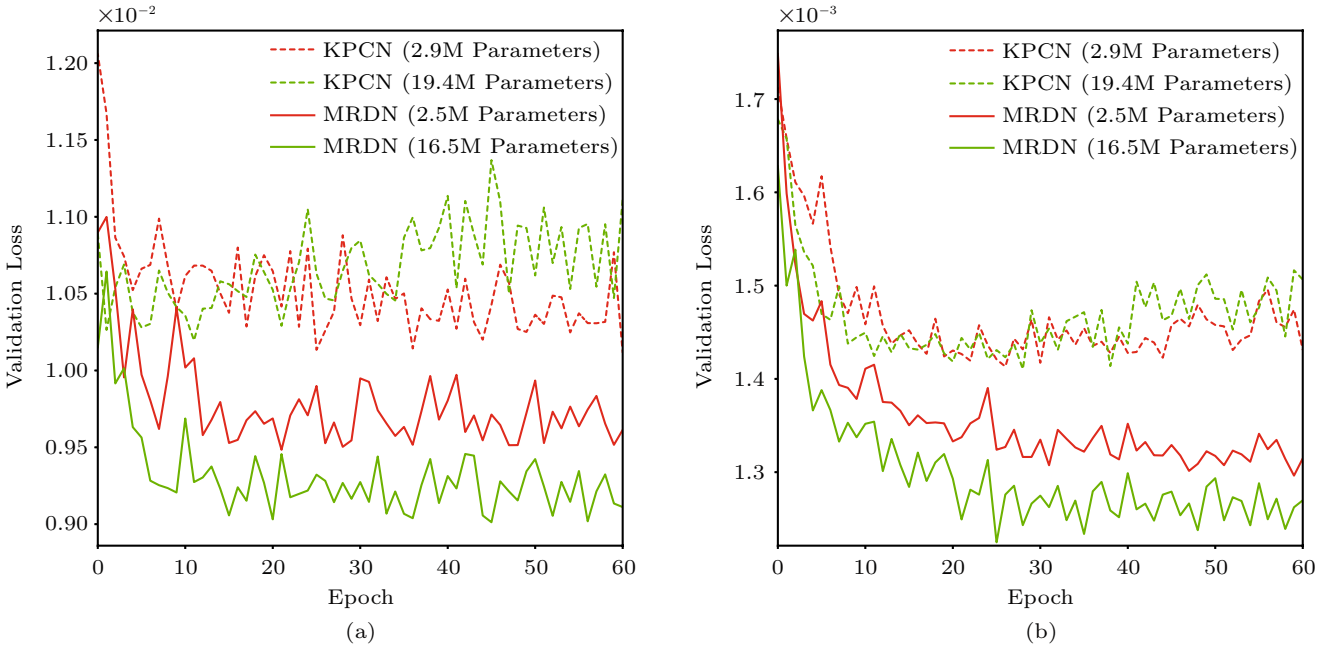

Fig.12. We further increase the trainable parameters to fully test the networks potential performance. As the number of parameters increased from 2.9 million to 19.4 million, KPCN becomes very unstable and easy to over-fit. But MRDN (ours) could gain a huge performance boost with the increasing number of parameters. (a) Convergence plot on global component. (b) Convergence plot on caustic component.

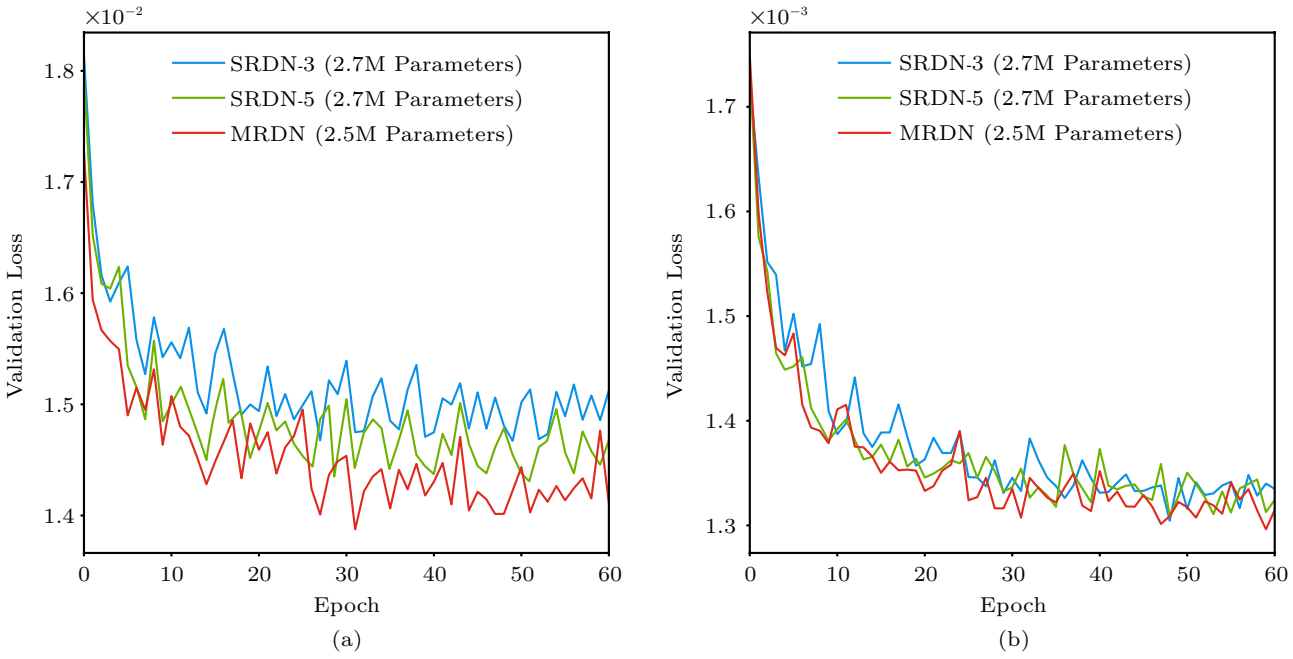

Fig. 13. Comparison of validation loss between the SRDNs and MRDN for both (a) the global and (b) the caustic components. Although MRDN has the lowest number of parameters, compared with SRDNs, it still achieves the best performance in both the global and caustic components. And in the global component, the advantage of MRDN is even more obvious.

faces. Meanwhile, compared with MRDN and SRDN-3, SRDN-5 finds it difficult to handle noises without over-smoothing on narrow regions with dramatic changes in geometry, like the boundary of walls and the cabinet seams. MRDN could remove multi-scale noises on both low-frequency and high-frequency areas, and as described in Subsection 3.2, it could gain benefits from both large and small sizes of filters.

Photon-Related Features Validation. To validate our photon-related features (photon density, flux, and gra-

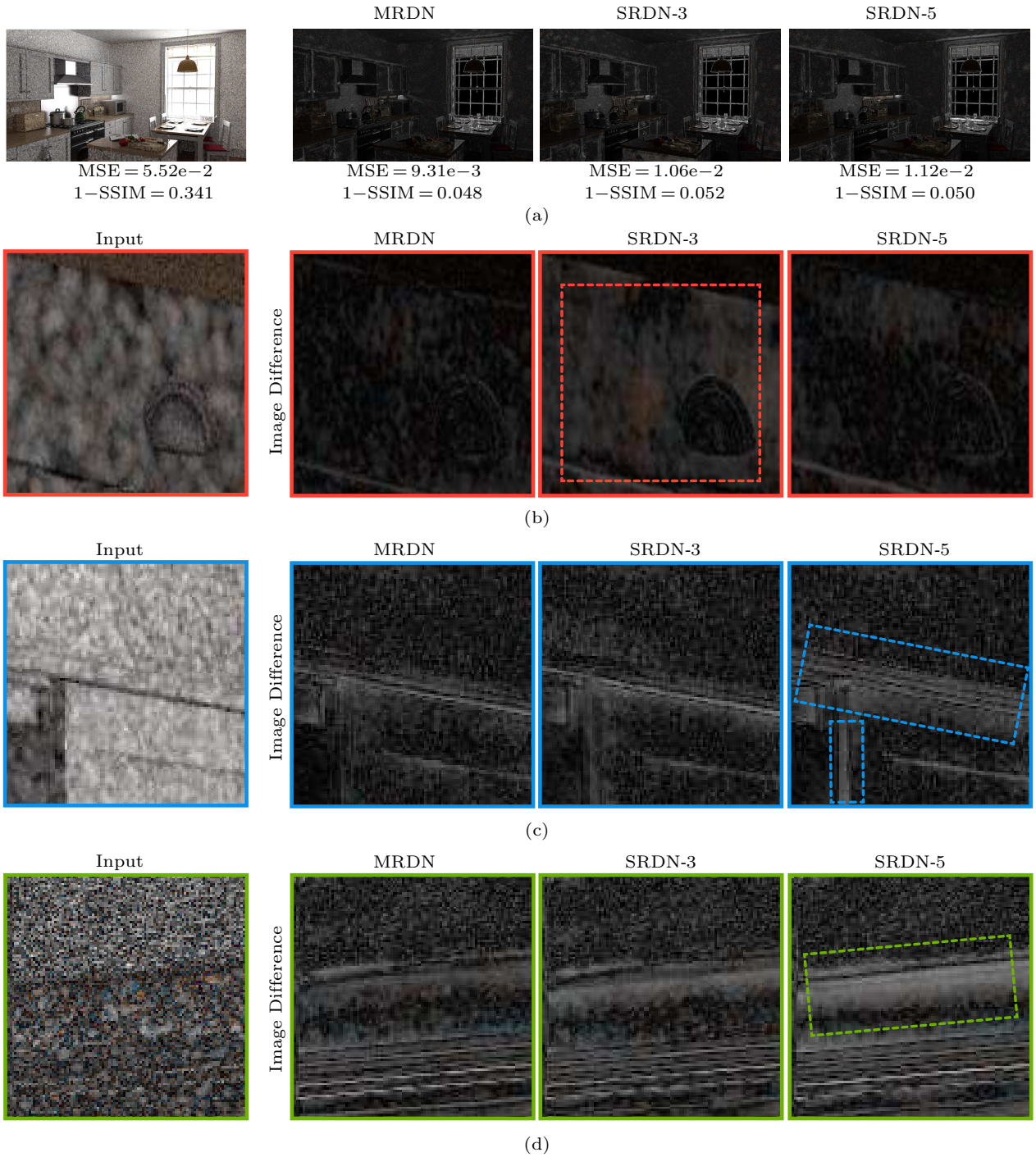

Fig.14. Image difference of references and denoising results of MRDN and SRDNs. Image difference equals the absolute value of the difference between the reference and the denoised image. The darker, the better. On the low-frequency area (see (b), the flat cupboard surface), MRDN and SRDN-5 can better remove the large noises. And on the high-frequency areas (see (c) and (d), the cupboard seams and the boundary of walls respectively), MRDN and SRDN-3 can better remove noises while avoiding over-smoothing. (a) Comparison of the whole image. (b) Close-up comparison of the cupboard surface. (c) Close-up comparison of the cupboard seams. (d) Close-up comparison of the boundary of walls.

dient), we remove them from the input and train the network again. Fig.15 shows the impact of photon-related features on our network performance. With photon-related features, the caustic network converges faster. We also show the impact of photon-related features on the denoising results in Fig.16. We observe that more illumination details, especially caustic, are better preserved with the photon-related features.

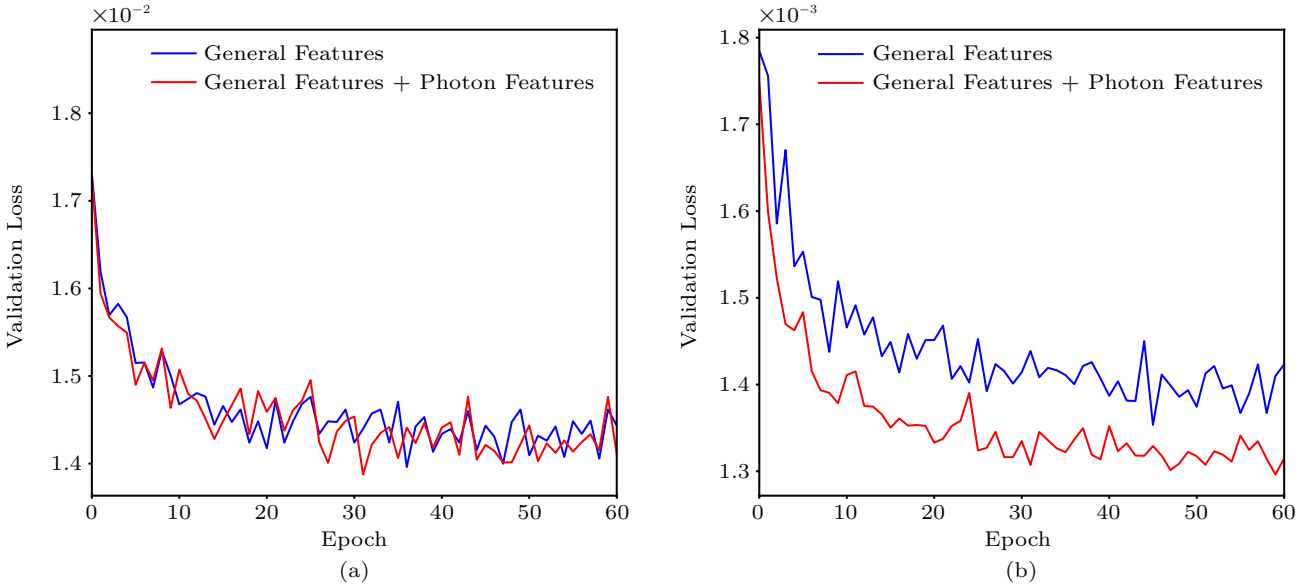

Fig.15. Impact of auxiliary photon-related features on (a) global and (b) caustic components via MRDN.

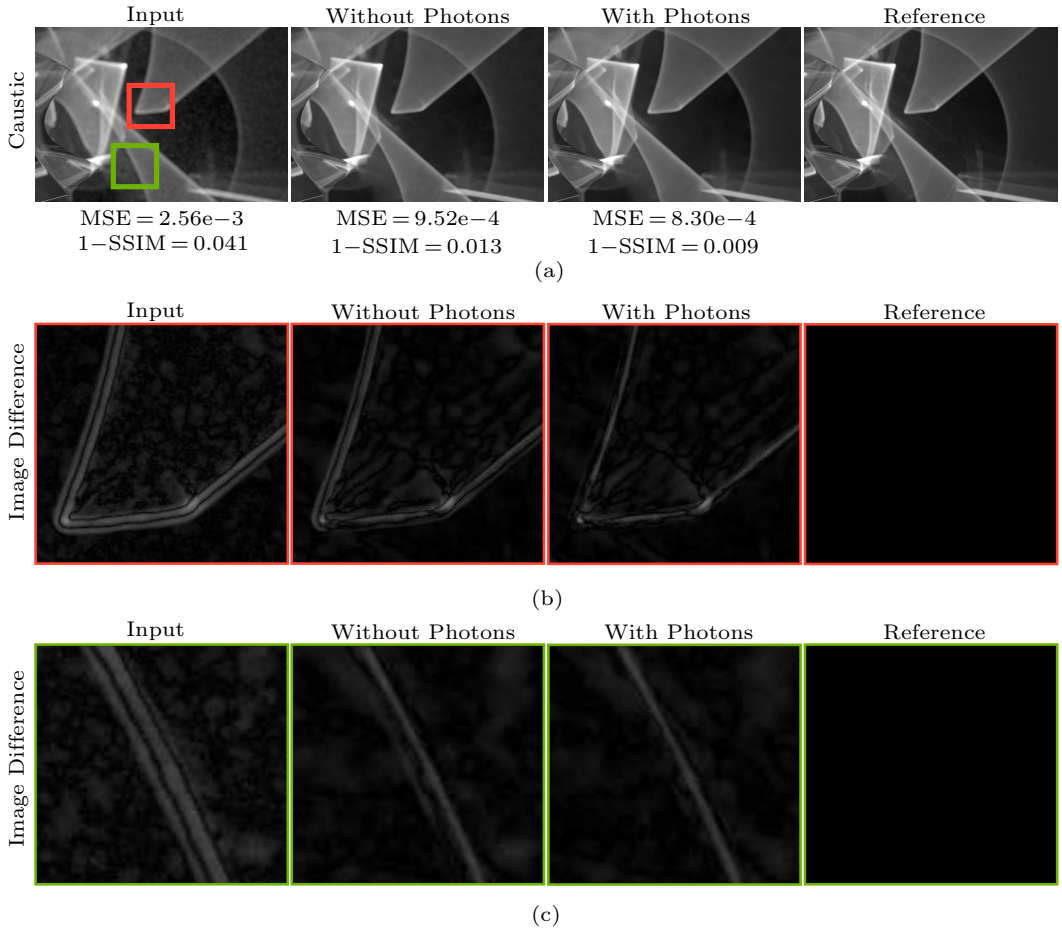

Fig. 16. Comparison of caustic component denoising result with and without photon-related features via MRDN. Image difference equals the absolute value of the difference between the reference and the denoised image. The darker, the better. With photon-related features, MRDN could stay more sharp caustics after denoising. (a) Comparison of the whole image. (b) Close-up comparison of the group of caustics. (c) Close-up comparison of the another group of caustics.

7 Conclusions

We presented the first learning-based image-space method for biased SPPM denoising. It can handle both bias and variance in SPPM renderings well at the same time, via a two-network framework. Specifically, we presented the multi-residual denoising network (MRDN) with a multi-residual block, which can benefit from both large and small sizes of filters. It allows better dealing with multi-scale noises on both low-frequency and high-frequency areas, and making MRDN more suitable for SPPM denoising task. We also proposed a series of auxiliary photon-related features to better handle noises while preserving correct illumination details, especially caustics. Compared with other learning-based MC denoising methods that we applied to SPPM denoising problem, our method can achieve higher denoising quality and preserve illumination details much better.

There are still some interesting improving directions in the future. Although our method performs well in our denoising tests, it could not handle extreme large noises when they are very different from large noises that appear in our training dataset. This limits the flexibility of our approach. Furthermore, we focus on single image denoising, and it would be useful to expand our framework to handle animated sequences rendered with SPPM.

References

[1] Hachisuka T, Jensen H W. Stochastic progressive photon mapping. *ACM Transactions on Graphics*, 2009, 28(5): Article No. 141.

[2] Jensen H W. Realistic Image Synthesis Using Photon Mapping (1st edition). Routledge, 2001.

[3] Hachisuka T, Ogaki S, Jensen H W. Progressive photon mapping. *ACM Transactions on Graphics*, 2008, 27(5): Article No. 130.

[4] Kang C, Wang L, Xu Y *et al*. A survey of photon mapping state-of-the-art research and future challenges. *Frontiers of Information Technology & Electronic Engineering*, 2016, 17(3): 185-199.

[5] Ritschel T, Dachsbacher C, Grosch T *et al*. The state of the art in interactive global illumination. *Computer Graphics Forum*, 2012, 31(1): 160-188.

[6] Bako S, Vogels T, McWilliams B *et al*. Kernel-predicting convolutional networks for denoising Monte Carlo renderings. *ACM Transactions on Graphics*, 2017, 36(4): Article No. 97.

[7] Vogels T, Rousselle F, McWilliams B *et al*. Denoising with kernel prediction and asymmetric loss functions. *ACM Transactions on Graphics*, 2018, 37(4): Article No. 124.

[8] Wong K M, Wong T T. Robust deep residual denoising for Monte Carlo rendering. In *Proc. the 2018 SIGGRAPH Asia Technical Briefs*, December 2018, Article No. 14.

[9] Wong K M, Wong T T. Deep residual learning for denoising Monte Carlo renderings. *Computational Visual Media*, 2019, 5(3): 239-255.

[10] Kalantari N K, Bako S, Sen P. A machine learning approach for filtering Monte Carlo noise. *ACM Transactions on Graphics*, 2015, 34(4): Article No. 122.

[11] Xu B, Zhang J, Wang R *et al*. Adversarial Monte Carlo denoising with conditioned auxiliary feature modulation. *ACM Transactions on Graphics*, 2019, 38(6): Article No. 224.

[12] Yang X, Wang D, Hu W *et al*. DEMC: A deep dual-encoder network for denoising Monte Carlo rendering. *Journal of Computer Science and Technology*, 2019, 34(5): 1123-1135.

[13] Gharbi M, Li T M, Aittala M *et al*. Sample-based Monte Carlo denoising using a kernel-splatting network. *ACM Transactions on Graphics*, 2019, 38(4): Article No. 125.

[14] Günther T, Grosch T. Distributed out-of-core stochastic progressive photon mapping. *Computer Graphics Forum*, 2014, 33(6): 154-166.

[15] Havran V, Bittner J, Herzog R *et al*. Ray maps for global illumination. *Rendering Techniques*, 2005, 2005: 43-54.

[16] Frisvad J R, Schjøth L, Erleben K *et al*. Photon differential splatting for rendering caustics. *Computer Graphics Forum*, 2014, 33(6): 252-263.

[17] Spencer B, Jones M W. Into the blue: Better caustics through photon relaxation. *Computer Graphics Forum*, 2009, 28(2): 319-328.

[18] Fu Z, Jensen H W. Noise reduction for progressive photon mapping. In *Proc. the International Conference on Computer Graphics and Interactive Techniques*, August 2012, Article No. 29.

[19] Kaplanyan A S, Dachsbacher C. Adaptive progressive photon mapping. *ACM Transactions on Graphics*, 2013, 32(2): Article No. 16.

[20] Bengio Y, Simard P, Frasconi P. Learning long-term dependencies with gradient descent is difficult. *IEEE Transactions on Neural Networks*, 1994, 5(2): 157-166.

[21] He K, Zhang X, Ren S *et al*. Identity mappings in deep residual networks. In *Proc. the 14th European Conference on Computer Vision*, October 2016, pp.630-645.

[22] He K, Zhang X, Ren S *et al*. Deep residual learning for image recognition. In *Proc. the 2016 IEEE Conference on Computer Vision and Pattern Recognition*, June 2016, pp.770-778.

[23] Zagoruyko S, Komodakis N. Wide residual networks. arXiv:1605.07146, 2016. https://arxiv.org/abs/1605.07146, Feb. 2019.

[24] Abdi M, Nahavandi S. Multi-residual networks: Improving the speed and accuracy of residual networks. arXiv:1609.05672, 2016. https://arxiv.org/abs/1609.05672, Feb. 2019.

[25] Silverman B W. Density Estimation for Statistics and Data Analysis. Chapman and Hall, 1986.

[26] Pharr M, Jakob W, Humphreys G. Physically Based Rendering: From Theory to Implementation (3rd edition). Morgan Kaufmann, 2016.

[27] He K, Zhang X, Ren S *et al*. Delving deep into rectifiers: Surpassing human-level performance on imageNet classification. In *Proc. the IEEE International Conference on Computer Vision*, December 2015, pp.1026-1034.

[28] Schjøth L, Sporring J, Olsen F O. Diffusion based photon mapping. *Computer Graphics Forum*, 2008, 27(8): 2114-2127.

[29] Abadi M, Barham P, Chen J *et al*. TensorFlow: A system for large-scale machine learning. In *Proc. the 12th USENIX Symposium on Operating Systems Design and Implementation*, November 2016, pp.265-283.

[30] Kingma D P, Ba J. Adam: A method for stochastic optimization. arXiv:1412.6980, 2014. https://arxiv.org/abs/1412.6980, Dec. 2019.

[31] Glorot X, Bengio Y. Understanding the difficulty of training deep feed forward neural networks. In *Proc. the 13th International Conference on Artificial Intelligence and Statistics*, May 2010, pp.249-256.

Zheng Zeng received his B.S. degree in digital media technology from Shandong University, Jinan, in 2018. He is currently a Master student in the School of Software at Shandong University, Jinan, supervised by Prof. Lu Wang. His research interests include photorealistic rendering and high-performance rendering.

Lu Wang is a professor at the School of Software, Shandong University, Jinan. She received her Ph.D. degree in computer science from Shandong University, Jinan, in 2009. Her research interests include photorealistic rendering and high-performance rendering.

Bei-Bei Wang is an associate professor at Nanjing University of Science and Technology (NJUST), Nanjing. She received her Ph.D. degree in computer science from Shandong University, Jinan, in 2014, and visited Telecom ParisTech from 2012 to 2014. She worked as a postdoctoral researcher in INRIA from 2015 to 2017. She joined NJUST in March 2017. Her research interests include rendering and game development.

Chun-Meng Kang received her B.S. degree in software engineering from Shandong University, Jinan, in 2011, and her Ph.D. degree in computer science from Shandong University, Jinan, in 2016. She is currently a lecturer in School of Information Science and Engineering, Shandong Normal University, Jinan. Her research interests include photorealistic rendering and high-performance computing.

Yan-Ning Xu is an associate professor at the School of Software, Shandong University, Jinan. He received his Ph.D. degree in computer science from Shandong University, Jinan, in 2006. His research interests include photorealistic rendering, high-performance rendering, virtual reality, and human-computer interaction.