Learning Camera-Aware Noise Models

Ke-Chi Chang[1,2][0000−0002−5500−1479], Ren Wang[1][0000−0001−5168−2337],
Hung-Jin Lin[1][0000−0002−5077−6320], Yu-Lun Liu[1][0000−0002−7561−6884],
Chia-Ping Chen[1][0000−0002−7022−3061], Yu-Lin Chang[1][0000−0002−2840−0052], and
Hwann-Tzong Chen[2][0000−0003−2806−7090]

[1] MediaTek Inc., Hsinchu, Taiwan
[2] National Tsing Hua University, Hsinchu, Taiwan

Abstract. Modeling imaging sensor noise is a fundamental problem for image processing and computer vision applications. While most previous works adopt statistical noise models, real-world noise is far more complicated and beyond what these models can describe. To tackle this issue, we propose a data-driven approach, where a generative noise model is learned from real-world noise. The proposed noise model is camera-aware, that is, different noise characteristics of different camera sensors can be learned simultaneously, and a single learned noise model can generate different noise for different camera sensors. Experimental results show that our method quantitatively and qualitatively outperforms existing statistical noise models and learning-based methods. The source code and more results are available at `https://arcchang1236.github.io/CA-NoiseGAN/`.

Keywords: Noise Model, Denoising, GANs, Sensor

1 Introduction

Modeling imaging sensor noise is an important task for many image processing and computer vision applications. Besides low-level applications such as image denoising [4,9,25,26], many high-level applications, such as detection or recognition [10,17,20,21], can benefit from a better noise model.

Many existing works assume statistical noise models in their applications. The most common and simplest one is signal-independent additive white Gaussian noise (AWGN) [25]. A combination of Poisson and Gaussian noise, containing both signal-dependent and signal-independent noise, is shown to be a better fit for most camera sensors [7,9].

However, the behavior of real-world noise is very complicated. Different noise can be induced at different stages of an imaging pipeline. Real-world noise includes but is not limited to photon noise, read noise, fixed-pattern noise, dark current noise, row/column noise, and quantization noise. Thus simple statistical noise models can not well describe the behavior of real-world noise.

Recently, several learning-based noise models are proposed to better represent the complexity of real-world noise in a data-driven manner [1,5,12]. In this paper,

we propose a learning-based generative model for signal-dependent synthetic noise. The synthetic noise generated by our model is perceptually more realistic than existing statistical models and other learning-based methods. When used to train a denoising network, better denoising quality can also be achieved.

Moreover, the proposed method is camera-aware. Different noise characteristics of different camera sensors can be learned simultaneously by a single generative noise model. Then this learned noise model can generate different synthetic noise for different camera sensors respectively.

Our main contributions are summarized as follows:

- propose a learning-based generative model for camera sensor noise
- achieve camera awareness by leveraging camera-specific Poisson-Gaussian noise and a camera characteristics encoding network
- design a novel feature matching loss for signal-dependent patterns, which leads to significant improvement of visual quality
- outperform state-of-the-art noise modeling methods and improve image denoising performance

2   Related Work

Image denoising is one of the most important applications and benchmarks in noise modeling. Similar to the recent success of deep learning in many vision tasks, deep neural networks also dominate recent advances of image denoising.

DnCNN [25] shows that a residual neural network can perform blind denoising well and obtains better results than previous methods on additive white Gaussian noise (AWGN). However, a recent denoising benchmark DND [19], consisting of real photographs, found that the classic BM3D method [6] outperforms DnCNN on real-world noise instead. The main reason is that real-world noise is more complicated than AWGN, and DnCNN failed to generalize to real-world noise because it was trained only with AWGN.

Instead of AWGN, CBDNet [9] and Brooks *et al.* [4] adopt Poisson-Gaussian noise and demonstrate significant improvement on the DND benchmark. Actually, they adopt an approximated version of Poisson-Gaussian noise by a heteroscedastic Gaussian distribution:

$$n \sim \mathcal{N}(0, \delta_{\text{shot}} I + \delta_{\text{read}}),\tag{1}$$

where n is the noise sampling, I is the intensity of a noise-free image, and δ_{shot} and δ_{read} denote the Poisson and Gaussian components, respectively. Moreover, δ_{shot} and δ_{read} for a specific camera sensor can be obtained via a calibration process [14]. The physical meaning of these two components corresponds to the signal-dependent and signal-independent noise of a specific camera sensor.

Recently, several learning-based noise modeling approaches have been proposed [1,5,12]. GCBD [5] is the first GAN-based noise modeling method. Its generative noise model, however, takes only a random vector as input but does not take the intensity of the clean image into account. That means the generated

noise is not signal-dependent. Different characteristics between different camera sensors are not considered either. The synthetic noise is learned and imposed on sRGB images, rather than the raw images. These are the reasons why GCBD didn't deliver promising denoising performance on the DND benchmark [19].

GRDN [12] is another GAN-based noise modeling method. Their model was trained with paired data of clean images and real noisy images of smartphone cameras, provided by NTIRE 2019 Real Image Denoising Challenge [3], which is a subset of the SIDD benchmark [2]. In addition to a random seed, the input of the generative noise model also contained many conditioning signals: the noise-free image, an identifier indicating the camera sensor, ISO level, and shutter speed. Although GRDN can generate signal-dependent and camera-aware noise, the denoising network trained with this generative noise model only improved slightly. The potential reasons are two-fold: synthetic noise was learned and imposed on sRGB images, not raw images; a plain camera identifier is too simple to represent noise characteristics of different camera sensors.

Noise Flow [1] applied a flow-based generative model that maximizes the likelihood of real noise on raw images, and then exactly evaluated the noise modeling performance qualitatively and quantitatively. To do so, the authors proposed using Kullback-Leibler divergence and negative log-likelihood as the evaluation metrics. Both training and evaluation were conducted on SIDD [2]. To our knowledge, Noise Flow is the first deep learning-based method that demonstrates significant improvement in both noise modeling and image denoising capabilities. However, they also fed only a camera identifier into a gain layer to represent complex noise characteristics of different camera sensors.

3   Proposed Method

Different from most existing works, the proposed learning-based approach aims to model noise characteristics for each camera sensor. Fig. 1 shows an overview of our framework, which comprises two parts: the Noise-Generating Network and the Camera-Encoding Network. The Noise-Generating Network, introduced in Sec. 3.1, learns to generate synthetic noise according to the content of a clean input image and the characteristics of a target camera. The target camera characteristics are extracted via the Camera-Encoding Network from noisy images captured by that target camera, which is illustrated in Sec. 3.2. Finally, Sec. 3.3 shows how to train these two networks in an end-to-end scheme.

3.1   Noise-Generating Network

As depicted in the upper part of Fig. 1, a clean image $\mathbf{I}_{C_i^s}$ from the sth camera and the initial synthetic noise $\tilde{\mathbf{n}}_{\mathrm{init}}$ are fed into a noise generator G and then transformed into various feature representations through convolutional layers. At the last layer, the network produces a residual image $R(\tilde{\mathbf{n}}_{\mathrm{init}}|\mathbf{I}_{C_i^s})$ that approximates the difference between real noise $\mathbf{n} \sim \mathbb{P}_r$ and $\tilde{\mathbf{n}}_{\mathrm{init}}$, where $\mathbb{P}_r$ indicates the
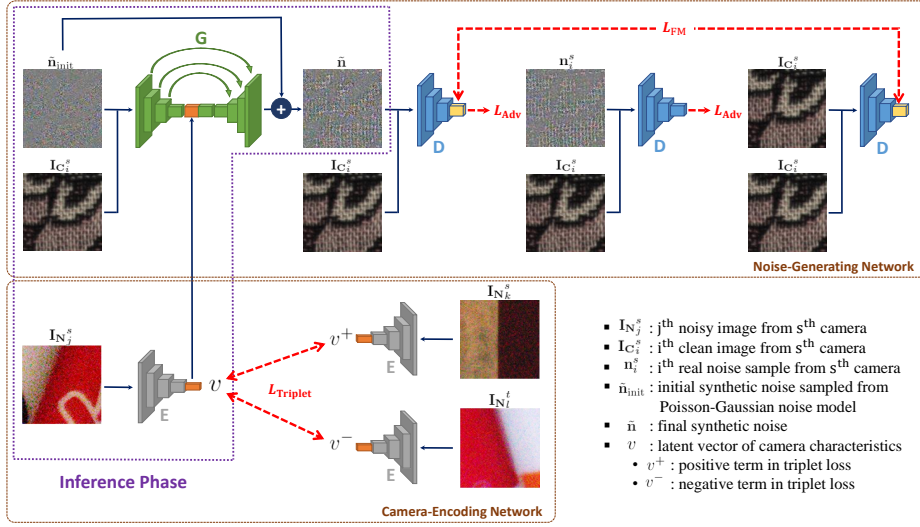
Fig. 1. An overview of our noise-modeling framework. The proposed architecture comprises two sub-networks: the Noise-Generating Network and the Camera-Encoding Network. First, a clean image $\mathbf{I}_{C_i^s}$ and the initial synthetic noise $\tilde{\mathbf{n}}_{\mathrm{init}}$ sampled from Poisson-Gaussian noise model are fed into the generator G. In addition, a latent vector $\boldsymbol{v}$ provided by the camera encoder E, which represents the camera characteristics, is concatenated with the features of the middle layers of G. Eventually, the final synthetic noise $\tilde{\mathbf{n}}$ is generated by G. To jointly train G and E, a discriminator D is introduced for the adversarial loss L_{Adv} and the feature matching loss L_{FM}. Moreover, a triplet loss L_{Triplet} is proposed to let the latent space of $\boldsymbol{v}$ be more reliable

real noise distribution. Ideally, we can generate realistic synthetic noise $\tilde{\mathbf{n}} \approx \mathbf{n}$ from the estimated residual image as

$$\tilde{\mathbf{n}} = G(\tilde{\mathbf{n}}_{\mathrm{init}}|\mathbf{I}_{C_i^s}) = \tilde{\mathbf{n}}_{\mathrm{init}} + R(\tilde{\mathbf{n}}_{\mathrm{init}}|\mathbf{I}_{C_i^s}). \tag{2}$$

To achieve this objective, we adopt adversarial learning for making the generated noise distribution $\mathbb{P}_g$ fit $\mathbb{P}_r$ as closely as possible. A discriminator D is used to measure the distance between distributions by distinguishing real samples from fake ones, such that G can minimize the distance through an adversarial loss L_{Adv}. Therefore, we need to collect pairs of clean images and real noise $(\mathbf{I}_{C_i^s}, \mathbf{n}_i^s)$ as the real samples.

A real noise sample $\mathbf{n}_i^s$ can be acquired by subtracting $\mathbf{I}_{C_i^s}$ from the corresponding noisy image $\mathbf{I}_{N_i^s}$, *i.e.*, $\mathbf{n}_i^s = \mathbf{I}_{N_i^s} - \mathbf{I}_{C_i^s}$. Note that a clean image could have many corresponding noisy images because noisy images can be captured at different ISOs to cover a wide range of noise levels. For simplicity, we let i denote not only the scene but also the shooting settings of a noisy image.

In addition to measuring the distance in adversarial learning, the discriminator D also plays another role in our framework. It is observed that some signal-dependent patterns like spots or stripes are common in real noise; hence

we propose a feature matching loss L_{FM} and treat D as a feature extractor. The feature matching loss forces the generated noise $\tilde{\mathbf{n}}$ and the clean image $\mathbf{I}_{C_i}^s$ to share similar high-level features because we assume these signal-dependent patterns should be the most salient traits in clean images.

It is worthwhile to mention that a noise model should be capable of generating a variety of reasonable noise samples for the same input image and noise level. GANs usually take a random vector sampled from Gaussian distribution as the input of the generator to ensure this stochastic property. In most cases, this random vector is not directly relevant to the main task. However, our goal is exactly to generate random noise, which implies that this random vector could be treated as the initial synthetic noise. Moreover, Gaussian distribution can be replaced with a more representative statistical noise model. For this reason, we apply Poisson-Gaussian noise model to the initial synthetic noise $\tilde{\mathbf{n}}_{\mathrm{init}}$ as in (1):

$$\tilde{\mathbf{n}}_{\mathrm{init}} \sim \mathcal{N}(0, \delta_{\mathrm{shot}_i}^s \mathbf{I}_{C_i}^s + \delta_{\mathrm{read}_i}^s), \tag{3}$$

where $\delta_{\mathrm{shot}_i}^s$ and $\delta_{\mathrm{read}_i}^s$ are the Poisson and the Gaussian component for $\mathbf{I}_{N_i}^s$, respectively. Note that these two parameters not only describe the preliminary noise model for the sth camera but also control the noise level of $\tilde{\mathbf{n}}_{\mathrm{init}}$ and $\tilde{\mathbf{n}}$.

3.2 Camera-Encoding Network

FUNIT [16] has shown that encoding the class information is helpful to specify the class domain for an input image. Inspired by their work, we would like to encode the camera characteristics in an effective representation. Since $\delta_{\mathrm{shot}_i}^s$ and $\delta_{\mathrm{read}_i}^s$ are related to the sth camera in (3), the generator G is actually aware of the camera characteristics from $\tilde{\mathbf{n}}_{\mathrm{init}}$. However, this awareness is limited to the assumption of the Poisson-Gaussian noise model. We, therefore, propose a novel Camera-Encoding Network to overcome this problem.

As depicted in the lower part of Fig. 1, a noisy image $\mathbf{I}_{N_j}^s$ is fed into a camera encoder E and then transformed into a latent vector $\boldsymbol{v} = E(\mathbf{I}_{N_j}^s)$. After that, the latent vector $\boldsymbol{v}$ is concatenated with the middle layers of G. Thus, the final synthetic noise is rewritten as

$$\tilde{\mathbf{n}} = G(\tilde{\mathbf{n}}_{\mathrm{init}}|\mathbf{I}_{C_i}^s, \boldsymbol{v}) = \tilde{\mathbf{n}}_{\mathrm{init}} + R(\tilde{\mathbf{n}}_{\mathrm{init}}|\mathbf{I}_{C_i}^s, \boldsymbol{v}). \tag{4}$$

We consider $\boldsymbol{v}$ as a representation for the characteristics of the sth camera and expect G can generate more realistic noise with this latent vector.

Aiming at this goal, the camera encoder E must have the ability to extract the core information for each camera, regardless of the content of input images. Therefore, a subtle but important detail here is that we feed the jth noisy image rather than the ith noisy image into E, whereas G takes the ith clean image as its input. Specifically, the jth noisy image is randomly selected from the data of the sth camera. Consequently, E has to provide latent vectors beneficial to the generated noise but ignoring the content of input images.

Additionally, some regularization should be imposed on $\boldsymbol{v}$ to make the latent space more reliable. FUNIT calculates the mean over a set of class images to

provide a representative class code. Nevertheless, this approach assumes that the latent space consists of hypersphere manifolds. Apart from FUNIT, we use a triplet loss L_{Triplet} as the regularization. The triplet loss is used to minimize the intra-camera distances while maximizing the inter-camera distances, which allows the latent space to be more robust to image content. The detailed formulation will be shown in the next section.

One more thing worth clarifying is why the latent vector $\boldsymbol{v}$ is extracted from the noisy image $\mathbf{I}_{N_j}^s$ rather than the real noise sample $\mathbf{n}_j^s$. The reason is out of consideration for making data preparation easier in the inference phase, which is shown as the violet block in Fig. 1. Collecting paired data $(\mathbf{I}_{C_j}^s, \mathbf{I}_{N_j}^s)$ to acquire $\mathbf{n}_j^s$ is cumbersome and time-consuming in real world. With directly using noisy images to extract latent vectors, there is no need to prepare a large number of paired data during the inference phase.

3.3   Learning

To jointly train the aforementioned networks, we have briefly introduced three loss functions: 1) the adversarial loss L_{Adv}, 2) the feature matching loss L_{FM}, and 3) the triplet loss L_{Triplet}. In this section, we describe the formulations for these loss functions in detail.

Adversarial Loss. GANs are well-known for reducing the divergence between the generated data distribution and real data distribution in the high-dimensional image space. However, there are several GAN frameworks for achieving this goal. Among these frameworks, we choose WGAN-GP [8] to calculate the adversarial loss L_{Adv}, which minimizes Wasserstein distance for stabilizing the training. The L_{Adv} is thus defined as

$$L_{\mathrm{Adv}} = - \mathop{\mathbb{E}}_{\tilde{\mathbf{n}} \sim \mathbb{P}_g} \left[D(\tilde{\mathbf{n}} | \mathbf{I}_C) \right], \tag{5}$$

where D scores the realness of the generated noise. In more depth, scores are given at the scale of patches rather than whole images because we apply a PatchGAN [11] architecture to D. The advantage of using this architecture is that it prefers to capture high-frequency information, which is associated with the characteristics of noise.

On the other hand, the discriminator D is trained by

$$L_D = \mathop{\mathbb{E}}_{\tilde{\mathbf{n}} \sim \mathbb{P}_g} \left[D(\tilde{\mathbf{n}} | \mathbf{I}_C) \right] - \mathop{\mathbb{E}}_{\mathbf{n} \sim \mathbb{P}_r} \left[D(\mathbf{n} | \mathbf{I}_C) \right] + \lambda_{\mathrm{gp}} \mathop{\mathbb{E}}_{\hat{\mathbf{n}} \sim \mathbb{P}_{\hat{\mathbf{n}}}} \left[(\|\nabla_{\hat{\mathbf{n}}} D(\hat{\mathbf{n}} | \mathbf{I}_C)\|_2 - 1)^2 \right], \tag{6}$$

where λ_{gp} is the weight of gradient penalty, and $\mathbb{P}_{\hat{\mathbf{n}}}$ is the distribution sampling uniformly along straight lines between paired points sampled from $\mathbb{P}_g$ and $\mathbb{P}_r$.

Feature Matching Loss. In order to regularize the training for GANs, some works [16,23] apply the feature matching loss and extract features through the discriminator networks. Following these works, we propose a feature matching

loss L_{FM} to encourage G to generate signal-dependent patterns in synthetic noise. The L_{FM} is then calculated as

$$L_{\text{FM}} = \underset{\tilde{\mathbf{n}} \sim \mathbb{P}_g}{\mathbb{E}} \left[\left\| D_f(\tilde{\mathbf{n}}|\mathbf{I}_C) - D_f(\mathbf{I}_C|\mathbf{I}_C) \right\|_1 \right], \tag{7}$$

where D_f denotes the feature extractor constructed by removing the last layer from D. Note that D_f is not optimized by L_{FM}.

Triplet Loss. The triplet loss was first proposed to illustrate the triplet relation in embedding space by [22]. We use the triplet loss to let the latent vector $\boldsymbol{v} = E(\mathbf{I}_{N_j^s})$ be more robust to the content of noisy image. Here we define the positive term $\boldsymbol{v}^+$ as the latent vector also extracted from the sth camera, and the negative term $\boldsymbol{v}^-$ is from a different camera on the contrary. In particular, $\boldsymbol{v}^+$ and $\boldsymbol{v}^-$ are obtained by encoding the randomly selected noisy images $\mathbf{I}_{N_k^s}$ and $\mathbf{I}_{N_l^t}$, respectively. Note that $\mathbf{I}_{N_k^s}$ is not restricted to any shooting setting, which means the images captured with different shooting settings of the same camera are treated as positive samples. The objective is to minimize the intra-camera distances while maximizing the inter-camera distances. The triplet loss L_{Triplet} is thus given by

$$L_{\text{Triplet}} = \underset{\boldsymbol{v},\boldsymbol{v}^+,\boldsymbol{v}^- \sim \mathbb{P}_e}{\mathbb{E}} \left[\max(0, \left\| \boldsymbol{v} - \boldsymbol{v}^+ \right\|_2 - \left\| \boldsymbol{v} - \boldsymbol{v}^- \right\|_2 + \alpha) \right], \tag{8}$$

where $\mathbb{P}_e$ is the latent space distribution and α is the margin between positive and negative pairs.

Full Loss. The full objective of the generator G is combined as

$$L_{\text{G}} = L_{\text{Adv}} + \lambda_{\text{FM}} L_{\text{FM}} + \lambda_{\text{Triplet}} L_{\text{Triplet}}, \tag{9}$$

where λ_{FM} and λ_{Triplet} control the relative importance for each loss term.

4   Experimental Results

In this section, we first describe our experiment settings and the implementation details. Then, Sec. 4.1 shows the quantitative and qualitative results. Sec. 4.2 presents extensive ablation studies to justify our design choices. The effectiveness and robustness of the Camera-Encoding Network are evaluated in Sec. 4.3.

Dataset. We train and evaluate our method on Smartphone Image Denoising Dataset (SIDD) [2], which consists of approximately 24,000 pairs of real noisy-clean images. The images are captured by five different smartphone cameras: Google Pixel, iPhone 7, Samsung Galaxy S6 Edge, Motorola Nexus 6, and LG G4. These images are taken in ten different scenes and under a variety of lighting conditions and ISOs. SIDD is currently the most abundant dataset available for real noisy and clean image pairs.

Table 1. Quantitative evaluation of different noise models. Our proposed noise model yields the best Kullback-Leibler divergence (D_{KL}). Relative improvements of our method over other baselines are shown in parentheses

	Gaussian	Poisson-Gaussian	Noise Flow	Ours
D_{KL}	0.54707 (99.5%)	0.01006 (74.7%)	0.00912 (72.0%)	**0.00159**

Implementation details. We apply Bayer preserving augmentation [15] to all SIDD images, including random cropping and horizontal flipping. At both training and testing phases, the images are cropped into 64×64 patches. Totally 650,000 pairs of noisy-clean patches are generated. Then we randomly select 500,000 pairs as the training set and 150,000 pairs as the test set. The scenes in the training set and test set are mutually exclusive to prevent overfitting. Specifically, the scene indices of the test set are 001, 002 and 008, and the remaining indices are used for the training set.

To synthesize the initial synthetic noise $\tilde{\mathbf{n}}_{\mathrm{init}}$, we set the Poisson component $\delta_{\mathrm{shot}_i}^s$ and Gaussian component $\delta_{\mathrm{read}_i}^s$ in (3) to the values provided by SIDD, which are estimated using the method proposed by [14]. The weight of gradient penalty of L_D in (6) is set to $\lambda_{\mathrm{gp}} = 10$, and the margin of L_{Triplet} in (8) is set to $\alpha = 0.2$. The loss weights of L_G in (9) are set to $\lambda_{\mathrm{FM}} = 1$ and $\lambda_{\mathrm{Triplet}} = 0.5$.

We use the Adam optimizer [13] in all of our experiments, with an initial learning rate of 0.0002, $\beta_1 = 0.5$, and $\beta_2 = 0.999$. Each training batch contains 64 pairs of noisy-clean patches. The generator G, discriminator D, and camera encoder E are jointly trained to convergence with 300 epochs. It takes about 3 days on a single GeForce GTX 1080 Ti GPU.

All of our experiments are conducted on linear raw images. Previous works have shown that many image processing methods perform better in Bayer RAW domain than in sRGB domain [19]. For noise modeling or image denoising, avoiding non-linear transforms (such as gamma correction) or spatial operations (such as demosaicking) is beneficial because we can prevent noise characteristics from being dramatically changed by these operations.

Methods in comparison. We compare our method with two mostly-used statistical models: Gaussian noise model and Poisson-Gaussian noise model, and one state-of-the-art learning-based method: Noise Flow [1].

4.1   Quantitative and qualitative results

To perform the quantitative comparison, we adopt the Kullback-Leibler divergence (D_{KL}) as suggested in [1]. Table 1 shows the average D_{KL} between real noise and synthetic noise generated by different noise models. Our method achieves the smallest average Kullback-Leibler divergence, which means that our method can synthesize more realistic noise than existing methods.

Table 2. Ablation study of our model. L_{adv}: the adversarial loss, L_{FM}: the feature matching loss, E: the Camera-Encoding Network, L_{Triplet}: the triplet loss. The Kullback-Leibler divergence D_{KL} is measured in different settings

L_{Adv}	$\checkmark$	$\checkmark$	$\checkmark$	$\checkmark$
L_{FM}		$\checkmark$	$\checkmark$	$\checkmark$
E			$\checkmark$	$\checkmark$
L_{Triplet}				$\checkmark$
D_{KL}	0.01445	0.01374	0.01412	**0.00159**

Fig. 2 shows the synthetic noise generated in linear RAW domain by all noise models and then processed by the camera pipeline toolbox provided by SIDD [2]. Each two consecutive rows represent an image sample for different ISOs (indicated by a number) and different lighting conditions (L and N denote low and normal lighting conditions respectively). Our method can indeed generate synthetic noise that is more realistic and perceptually closer to the real noise.

4.2   Ablation Studies

In this section, we perform ablation studies to investigate how each component contributes to our method, including the feature matching loss L_{FM}, the Camera-Encoding Network E, the triplet loss L_{Triplet}, and the initial synthetic noise $\tilde{\mathbf{n}}_{\text{init}}$. The results are shown in Table 2 and 3.

Feature matching loss L_{FM}. Fig. 3 shows that the feature matching loss is effective in synthesizing signal-dependent noise patterns and achieving better visual quality. With the feature matching loss L_{FM}, the network is more capable of capturing low-frequency signal-dependent patterns. As shown in Table 2, the Kullback-Leibler divergence can also be improved from 0.01445 to 0.01374.

Camera-Encoding Network and triplet loss. The Camera-Encoding Network is designed to represent the noise characteristics of different camera sensors. However, simply adding a Camera-Encoding Network alone provides no advantage ($0.01374 \rightarrow 0.01412$), as shown in Table 2. The triplet loss is essential to learn effective camera-specific latent vectors, and the KL divergence can be significantly reduced from 0.01412 to 0.00159.

The camera-specific latent vectors can also be visualized in the t-SNE space [18]. As shown in Fig. 4, the Camera-Encoding Network can effectively extract camera-specific latent vectors from a single noisy image with the triplet loss.

Initial synthetic noise $\tilde{\mathbf{n}}_{\text{init}}$. Table 3 shows the average KL divergence when using Gaussian or Poisson-Gaussian noise as the initial noise $\tilde{\mathbf{n}}_{\text{init}}$.

Table 3. Ablation study of the initial synthetic noise $\tilde{\mathbf{n}}_{\text{init}}$ Using Poisson-Gaussian as initial synthetic noise model performs better than using Gaussian

$\tilde{\mathbf{n}}_{\text{init}}$	Gaussian	Poisson-Gaussian
D_{KL}	0.06265	**0.00159**

Table 4. Analysis of noisy images from different cameras. The comparison of Kullback-Leibler divergence for different cameras of the noisy image, where $\tilde{\mathbf{n}}_A = G(\tilde{\mathbf{n}}_{\text{init}}|\mathbf{I}_{C_i}^s, E(\mathbf{I}_{N_j}^s))$ and $\tilde{\mathbf{n}}_B = G(\tilde{\mathbf{n}}_{\text{init}}|\mathbf{I}_{C_i}^s, E(\mathbf{I}_{N_k}^t))$

	$(\tilde{\mathbf{n}}_A\|\mathbf{n}_i^s)$	$(\tilde{\mathbf{n}}_A\|\mathbf{n}_j^s)$	$(\tilde{\mathbf{n}}_B\|\mathbf{n}_i^s)$
D_{KL}	0.00159	0.17921	0.01324

The KL divergence severely degrades from 0.00159 to 0.06265 if we use Gaussian noise instead of Poisson-Gaussian noise. This result shows that using a better synthetic noise as initial and predicting a residual to refine it can yield better-synthesized noise.

4.3   Robustness Analysis of the Camera-Encoding Network

To further verify the behavior and justify the robustness of the Camera-Encoding Network, we design several experiments with different input conditions.

Comparing noise for different imaging conditions or different cameras. Given a clean image $\mathbf{I}_{C_i}^s$ and a noisy image $\mathbf{I}_{N_j}^s$ also from the sth camera, our noise model should generate noise $\tilde{\mathbf{n}}_A = G(\tilde{\mathbf{n}}_{\text{init}}|\mathbf{I}_{C_i}^s, E(\mathbf{I}_{N_j}^s))$. The Kullback-Leibler divergence $D_{\text{KL}}(\tilde{\mathbf{n}}_A\|\mathbf{n}_i^s)$ between the generated noise and the corresponding real noise should be very small (0.00159 in Table 4). On the other hand, $D_{\text{KL}}(\tilde{\mathbf{n}}_A\|\mathbf{n}_j^s)$ between the generated noise and a non-corresponding real noise should be quite large (0.17921 in Table 4), owing to the different imaging conditions, even though the real noise $\mathbf{n}_j^s$ is from the same s^{th} camera.

If the latent vector is extracted by a noisy image of the tth camera instead of the sth camera, the generated noise becomes $\tilde{\mathbf{n}}_B = G(\tilde{\mathbf{n}}_{\text{init}}|\mathbf{I}_{C_i}^s, E(\mathbf{I}_{N_k}^t))$. Because the latent vector is from a different camera, we expect that $D_{\text{KL}}(\tilde{\mathbf{n}}_A\|\mathbf{n}_i^s) < D_{\text{KL}}(\tilde{\mathbf{n}}_B\|\mathbf{n}_i^s)$. Table 4 also verifies these results.

Analysis of different noisy images from the same camera. Another important property of the Camera-Encoding Network is that it must capture camera-specific characteristics from a noisy image, and the extracted latent vector should be irrelevant to the image content of the input noisy image. To verify this, we randomly select five different noisy images from the same camera. These

Table 5. Analysis of different noisy images from the same camera. The Kullback-Leibler divergence results from five randomly selected noisy images but fixed inputs for the generator

Noisy image sets	1st	2nd	3rd	4th	5th
D_{KL}	0.00159	0.00180	0.00183	0.00163	0.00176

different noisy images are fed into the Camera-Encoding Network, while other inputs for the Noise Generating Network are kept fixed. Because these noisy images are from the same camera, the generated noise should be robust and consistent. Table 5 shows that the D_{KL} between the generated noise and real noise remains low for different noisy images.

5   Application to Real Image Denoising

5.1   Real-world image denoising

We conduct real-world denoising experiments to further compare different noise models. For all noise models, we follow Noise Flow [1] to use the same 9-layer DnCNN network [25] as the baseline denoiser. Learning-based noise models (Noise Flow and ours) are trained with SIDD dataset. We then train a denoiser network with synthetic training pairs generated by each noise model separately.

Table 6 shows the average PSNR and SSIM [24] on the test set. The denoisers trained with statistical noise models (Gaussian and Poisson-Gaussian) are worse than those trained with learning-based noise models (Noise Flow and Ours), which also outperform the denoiser trained with real data only (the last row of Table 6). This is because the amount of synthetic data generated by noise models is unlimited, while the amount of real data is fixed.

Our noise model outperforms Noise Flow in terms of both PSNR and SSIM while using more training data for training noise models leads to better denoising performance. Table 6 also shows that using both real data and our noise model results in further improved PSNR and SSIM.

5.2   Camera-specific denoising networks

To verify the camera-aware ability of our method, we train denoiser networks with our generative noise models, which are trained with and without the Camera-Encoding Network (and with and without the triplet loss) respectively.

For our noise model without the Camera-Encoding Network, we train a single generic denoiser network for all cameras. For our noise models with the Camera-Encoding Network, we train camera-specific denoiser networks with and without the triplet loss for each camera. The denoising performance is shown in Table 7.

The results show that the Camera-Encoding Network with the triplet loss can successfully capture camera-specific noise characteristics and thus enhance the performance of camera-specific denoiser networks.

Table 6. Real-world image denoising. The denoising networks using our noise model outperform those using existing statistical noise models and learning-based models. **Red** indicates the best and <u>blue</u> indicates the second best performance (While training using both synthetic and real data, Ours + Real, synthetic and real data are sampled by a ratio of 5 : 1 in each mini-batch)

Noise model	# of training data for noise model	# of *real* training data for denoiser	PSNR	SSIM
Gaussian	-	-	43.63	0.968
Poisson-Gaussian	-	-	44.99	0.982
Noise Flow [1]	100k	-	47.49	0.991
	500k	-	48.52	0.992
Ours	100k	-	47.97	0.992
	500k	-	<u>48.71</u>	<u>0.993</u>
Ours + Real	100k	100k	47.93	**0.994**
	500k	500k	**48.72**	**0.994**
Real only	-	100k	47.08	0.989
	-	500k	48.30	**0.994**

Table 7. Real-world image denoising using camera-aware noise model. Grouping the proposed Camera-Encoding Network and triplet loss L_{Triplet} can extract camera-specific latent vectors and thus improve camera-specific denoiser networks

Model	PSNR on test cameras				
	IP	GP	S6	N6	G4
w/o $(E + L_{\text{Triplet}})$	57.4672	44.5180	40.0183	44.7954	51.8048
with E, w/o L_{Triplet}	49.8788	45.7755	40.4976	41.8447	51.8139
with $(E + L_{\text{Triplet}})$	**58.6073**	**45.9624**	**41.8881**	**46.4726**	**53.2610**

6   Conclusions

We have presented a novel learning-based generative method for real-world noise. The proposed noise model outperforms existing statistical models and learning-based methods quantitatively and qualitatively. Moreover, the proposed method can capture different characteristics of different camera sensors in a single noise model. We have also demonstrated that the real-world image denoising task can benefit from our noise model. As for future work, modeling real-world noise with few-shot or one-shot learning could be a possible direction. This could reduce the burden of collecting real data for training a learning-based noise model.
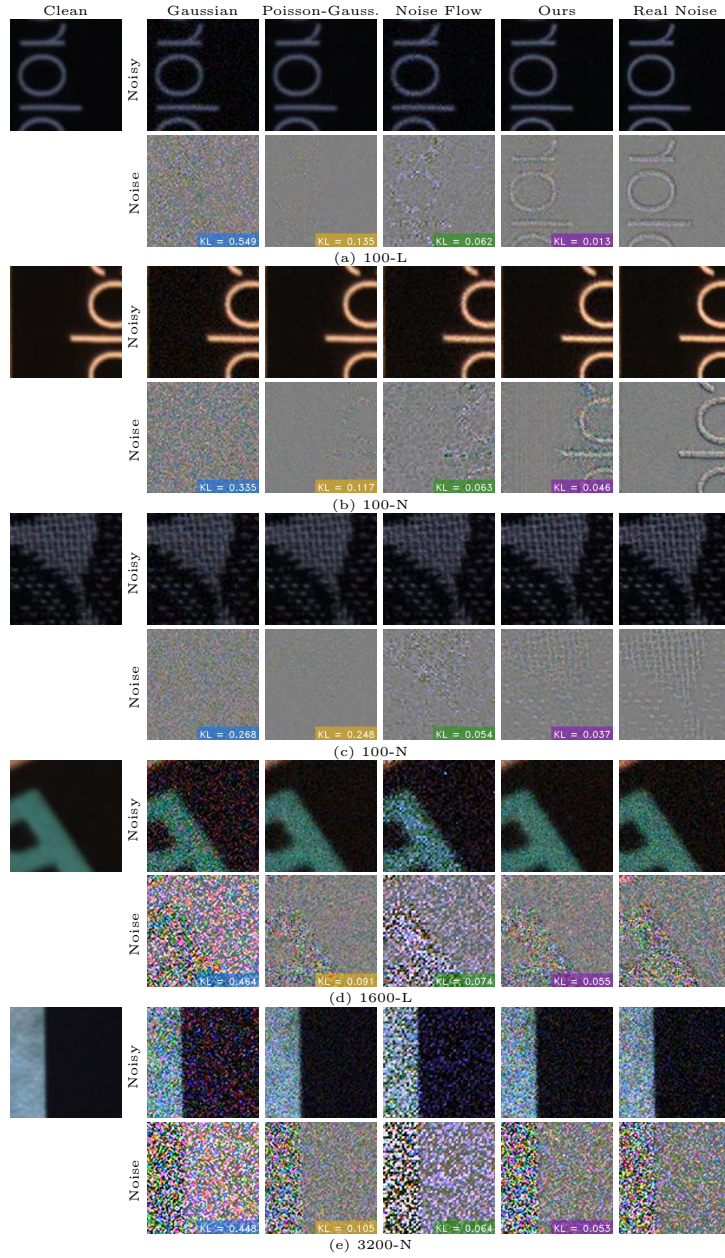
Fig. 2. Visualization of noise models. The synthetic noise samples of several noise modeling methods on different clean images with different ISO/lighting conditions are illustrated Quantitatively, the proposed method outperforms others in terms of D_{KL} measurement with real noise distribution. Furthermore, ours have many clear structures that fit the texture of clean images and real noise. Note that the noise value is scaled up for better visualization purpose
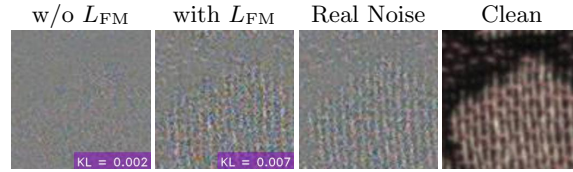
w/o L_{FM} with L_{FM} Real Noise   Clean

KL = 0.002   KL = 0.007

Fig. 3. Visualization of synthetic noise with and without feature matching loss. With the feature matching loss L_{FM}, the generated noise is highly correlated to the image content. Hence, they have more distinct structures resembling the texture of clean images. Besides, the D_{KL} measurements are slightly improved

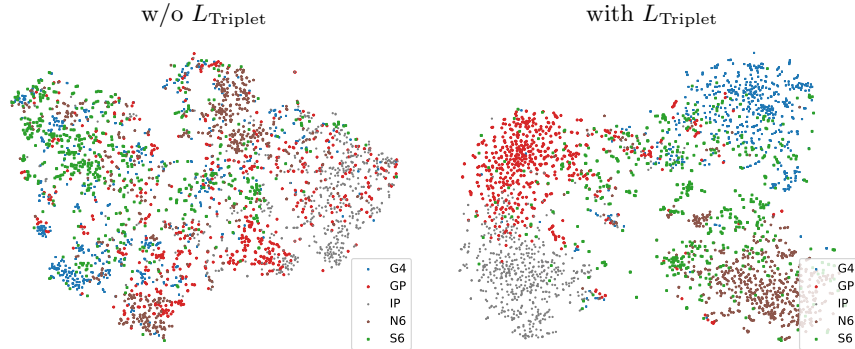w/o L_{Triplet} with L_{Triplet}

Fig. 4. Ablation study on the distributions of latent vectors from a camera encoder trained with or without L_{Triplet}. We project the encoded latent vectors v of noisy images from five different cameras with t-SNE. The Camera-Encoding Network trained with L_{Triplet} can effectively group the characteristics of different cameras

Real Noisy   Poisson-Gauss.   Real Only   Noise Flow   Ours   Ours + Real   Ground Truth

PSNR = 42.43   PSNR = 43.19   PSNR = 44.53   PSNR = 45.06   PSNR = 46.37

PSNR = 38.16   PSNR = 39.41   PSNR = 42.39   PSNR = 43.83   PSNR = 44.52

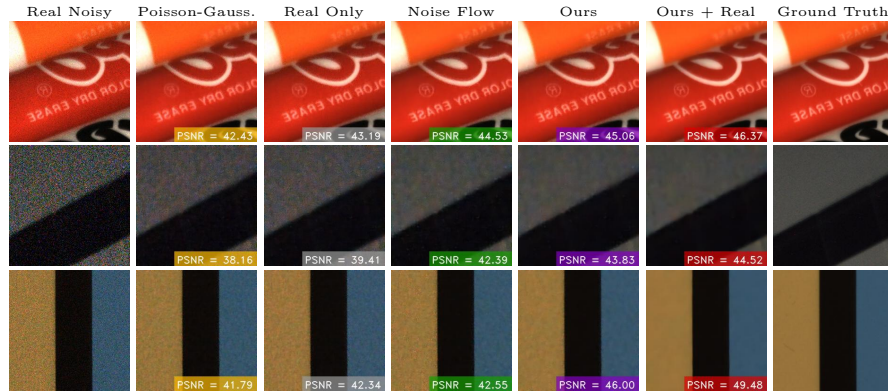PSNR = 41.79   PSNR = 42.34   PSNR = 42.55   PSNR = 46.00   PSNR = 49.48

Fig. 5. Results of denoisers trained on different noise models. We compare the denoised results trained on different settings, 1) only real pairs, 2) synthetic pairs with Noise Flow or the proposed method, and 3) the mixture of synthetic pairs from ours and real pairs

References

1. Abdelhamed, A., Brubaker, M.A., Brown, M.S.: Noise flow: Noise modeling with conditional normalizing flows. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 3165–3173 (2019)
2. Abdelhamed, A., Lin, S., Brown, M.S.: A high-quality denoising dataset for smartphone cameras. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 1692–1700 (2018)
3. Abdelhamed, A., Timofte, R., Brown, M.S., et al.: Ntire 2019 challenge on real image denoising: Methods and results. In: The IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW) (June 2019)
4. Brooks, T., Mildenhall, B., Xue, T., Chen, J., Sharlet, D., Barron, J.T.: Unprocessing images for learned raw denoising. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 11036–11045 (2019)
5. Chen, J., Chen, J., Chao, H., Yang, M.: Image blind denoising with generative adversarial network based noise modeling. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 3155–3164 (2018)
6. Dabov, K., Foi, A., Katkovnik, V., Egiazarian, K.: Image denoising with block-matching and 3d filtering. In: Image Processing: Algorithms and Systems, Neural Networks, and Machine Learning. vol. 6064, p. 606414. International Society for Optics and Photonics (2006)
7. Foi, A., Trimeche, M., Katkovnik, V., Egiazarian, K.: Practical poissonian-gaussian noise modeling and fitting for single-image raw-data. Trans. Img. Proc. **17**(10), 1737–1754 (Oct 2008). https://doi.org/10.1109/TIP.2008.2001399, `https://doi.org/10.1109/TIP.2008.2001399`
8. Gulrajani, I., Ahmed, F., Arjovsky, M., Dumoulin, V., Courville, A.C.: Improved training of wasserstein gans. In: Advances in neural information processing systems. pp. 5767–5777 (2017)
9. Guo, S., Yan, Z., Zhang, K., Zuo, W., Zhang, L.: Toward convolutional blind denoising of real photographs. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 1712–1722 (2019)
10. He, K., Gkioxari, G., Dollár, P., Girshick, R.: Mask r-cnn. In: Proceedings of the IEEE international conference on computer vision. pp. 2961–2969 (2017)
11. Isola, P., Zhu, J.Y., Zhou, T., Efros, A.A.: Image-to-image translation with conditional adversarial networks. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 1125–1134 (2017)
12. Kim, D.W., Ryun Chung, J., Jung, S.W.: Grdn: Grouped residual dense network for real image denoising and gan-based real-world noise modeling. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops. pp. 0–0 (2019)
13. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)
14. Liu, C., Szeliski, R., Kang, S.B., Zitnick, C.L., Freeman, W.T.: Automatic estimation and removal of noise from a single image. IEEE transactions on pattern analysis and machine intelligence **30**(2), 299–314 (2007)
15. Liu, J., Wu, C.H., Wang, Y., Xu, Q., Zhou, Y., Huang, H., Wang, C., Cai, S., Ding, Y., Fan, H., et al.: Learning raw image denoising with bayer pattern unification and bayer preserving augmentation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops. pp. 0–0 (2019)

16. Liu, M.Y., Huang, X., Mallya, A., Karras, T., Aila, T., Lehtinen, J., Kautz, J.: Few-shot unsupervised image-to-image translation. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 10551–10560 (2019)

17. Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.Y., Berg, A.C.: Ssd: Single shot multibox detector. In: European conference on computer vision. pp. 21–37. Springer (2016)

18. Maaten, L.v.d., Hinton, G.: Visualizing data using t-sne. Journal of machine learning research **9**(Nov), 2579–2605 (2008)

19. Plotz, T., Roth, S.: Benchmarking denoising algorithms with real photographs. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 1586–1595 (2017)

20. Redmon, J., Divvala, S., Girshick, R., Farhadi, A.: You only look once: Unified, real-time object detection. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 779–788 (2016)

21. Ren, S., He, K., Girshick, R., Sun, J.: Faster r-cnn: Towards real-time object detection with region proposal networks. In: Advances in neural information processing systems. pp. 91–99 (2015)

22. Schroff, F., Kalenichenko, D., Philbin, J.: Facenet: A unified embedding for face recognition and clustering. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 815–823 (2015)

23. Wang, T.C., Liu, M.Y., Zhu, J.Y., Tao, A., Kautz, J., Catanzaro, B.: High-resolution image synthesis and semantic manipulation with conditional gans. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 8798–8807 (2018)

24. Wang, Z., Bovik, A.C., Sheikh, H.R., Simoncelli, E.P.: Image quality assessment: from error visibility to structural similarity. IEEE transactions on image processing **13**(4), 600–612 (2004)

25. Zhang, K., Zuo, W., Chen, Y., Meng, D., Zhang, L.: Beyond a gaussian denoiser: Residual learning of deep cnn for image denoising. IEEE Transactions on Image Processing **26**(7), 3142–3155 (2017)

26. Zhang, K., Zuo, W., Zhang, L.: Ffdnet: Toward a fast and flexible solution for cnn-based image denoising. IEEE Transactions on Image Processing **27**(9), 4608–4622 (2018)