Supplementary Material
sRGB Real Noise Synthesizing with Neighboring Correlation-Aware Noise Model

Zixuan Fu[1]*, Lanqing Guo[1]*, Bihan Wen[1]†
[1]Nanyang Technological University, Singapore
{zixuan.fu, lanqing001, bihan.wen}@ntu.edu.sg

1. Network Architectures

1.1. Generator Network Architectures

Our proposed sRGB synthesizing framework contains three sub-networks: the gain estimation network (GENet), the noise-level prediction network (NPNet), and the neighboring correlation network (NCNet).

GENet. Table 1 shows the configuration of GENet, which contains five convolutional (conv) layers, a global average pooling layer [4], and a fully connected layer. The first layer is a 7×7 conv layer while the kernel size of the other four conv layers is 3×3. The activation function for each conv layer is ReLU. After the conv layers, a global average pooling layer and a fully connected layer are applied to the feature map of the fifth conv to obtain the estimated gain factor.

NPNet. Table 2 describes the architecture of the NPNet. The initial step involves embedding the clean patch into a 96 dimensional vector through a 7×7 conv layer. Next, three gain layers are applied to the feature map of the first layer, each consisting of a 1×1 convolution, a layer normalization (LN) [1], and a Leaky ReLU (LReLU) [5]. At each gain layer, the feature map is normalized by LN and multiplied by the gain factor. Finally, a 1×1 conv layer is employed to obtain the predicted noise level map.

NCNet. We adopt the widely used U-Net [6] architecture to map neighboring correlation onto the signal-dependent and neighboring uncorrelated (SDNU) noise, which is introduced in Table 3. For better optimizing the network, we apply a global residual connection [3] between the output of the NCNet and the input SDNU noise.

1.2. Discriminator Network Architectures

We adopt two discriminators, discriminator D_1 for NCNet and discriminator D_2 for NPNet, to score the realness of the synthesized noise. The network architectures of D_1 and D_2 are similar which both contain several conv layers for feature extraction and several fully connected layers to present the score of realness. The network architectures of two discriminators are shown in Table, respectively.

Inputs	Operator	Kernel	Output Channels	Stride	Output Size	Output Name
Noisy	Conv&ReLU	7×7	96	1	$h \times w$	GE_conv1
GE_conv1	Conv&ReLU	3×3	96	1	$h \times w$	GE_conv2
GE_conv2	Conv&ReLU	3×3	96	1	$h \times w$	GE_conv3
GE_conv3	Conv&ReLU	3×3	96	1	$h \times w$	GE_conv4
GE_conv4	Conv&ReLU	3×3	96	1	$h \times w$	GE_conv5
GE_conv5	Avg Pool	–	96	–	1×1	GE_pool1
GE_pool1	Conv	1×1	1	1	1×1	Gain_factor

Table 1. Architecture of the gain estimation network (GENet). Avg pool denotes the global average pooling layer [4].

Inputs	Operator	Kernel	Output Channels	Stride	Output Size	Output Name
Clean	Conv	7×7	96	1	$h \times w$	NP_conv1
NP_conv1	Conv	1×1	96	1	$h \times w$	NP_conv2
NP_conv2	LN	–	96	–	$h \times w$	NP_norm1
NP_norm1, Gain_factor	Pixel-wise multiplication	–	96	–	$h \times w$	NP_gain1
NP_gain1	LReLU	–	96	–	$h \times w$	NP_act1
NP_act1	Conv	1×1	96	1	$h \times w$	NP_conv3
NP_conv3	LN	–	96	–	$h \times w$	NP_norm2
NP_norm2, Gain_factor	Pixel-wise multiplication	–	96	–	$h \times w$	NP_gain2
NP_gain2	LReLU	–	96	–	$h \times w$	NP_act2
NP_act2	Conv	1×1	96	1	$h \times w$	NP_conv4
NP_conv4	LN	–	96	–	$h \times w$	NP_norm3
NP_norm3, Gain_factor	Pixel-wise multiplication	–	96	–	$h \times w$	NP_gain3
NP_gain3	LReLU	–	96	–	$h \times w$	NP_act3
NP_act3	Conv	1×1	3	1	$h \times w$	Noise_level_map

Table 2. Architecture of the noise-level prediction network (NPNet). LN denotes the layer normalization [1], and LReLU indicates the Leaky ReLU [5].

2. Pixel-Shuffle Down-Sampling Scheme

Directly calculating the adversarial loss $\mathcal{L}_{adv2}$ between the synthesized SDNU noise $\hat{\boldsymbol{v}}$ and real noise $\boldsymbol{n}$ is improper as the neighboring correlation of real noise. To address this issue, we adopt the Pixel-Shuffle Down-Sampling (PD) scheme proposed in [7], as illustrated in Figure 1, to reduce the neighboring correlation of real noise. PD first divides the noise map into $s \times s$ cells without overlapping, and then selects the pixels at the same location of each cells to create a down-sampled version. The PD scheme is applied on both the real noise and the SDNU noise $(\boldsymbol{n}, \hat{\boldsymbol{v}})$ to obtain their down-sampled versions $((\boldsymbol{n}) \downarrow_s, (\hat{\boldsymbol{v}}) \downarrow_s)$. According to [7], the neighboring correlation in the down-sampled version of

*Co-first authors contributed equally.
†Corresponding author: Bihan Wen.

Inputs	Operator	Kernel	Output Channels	Stride	Output Size	Output Name
SDNU_noise	Conv	7×7	64	1	$h \times w$	NC_conv1_1
NC_conv1_1	Conv&ReLU	3×3	64	1	$h \times w$	NC_conv1_2
NC_conv1_2	Conv&ReLU	3×3	128	2	$\frac{h}{2} \times \frac{w}{2}$	NC_conv2_1
NC_conv2_1	Conv&ReLU	3×3	128	1	$\frac{h}{2} \times \frac{w}{2}$	NC_conv2_2
NC_conv2_2	Conv&ReLU	3×3	256	2	$\frac{h}{4} \times \frac{w}{4}$	NC_conv3_1
NC_conv3_1	Conv&ReLU	3×3	256	1	$\frac{h}{4} \times \frac{w}{4}$	NC_conv3_2
NC_conv3_2	Conv&ReLU	3×3	512	2	$\frac{h}{8} \times \frac{w}{8}$	NC_conv4_1
NC_conv4_1	Conv&ReLU	3×3	512	1	$\frac{h}{8} \times \frac{w}{8}$	NC_conv4_2
NC_conv4_2	Upsampling	–	512	–	$\frac{h}{4} \times \frac{w}{4}$	NC_up1
NC_up1	Conv&ReLU	3×3	256	1	$\frac{h}{4} \times \frac{w}{4}$	NC_conv5_1
NC_conv5_1, NC_conv3_2	Concat	–	512	–	$\frac{h}{4} \times \frac{w}{4}$	NC_concat1
NC_concat1	Upsampling	–	512	–	$\frac{h}{2} \times \frac{w}{2}$	NC_up2
NC_up2	Conv&ReLU	3×3	128	1	$\frac{h}{2} \times \frac{w}{2}$	NC_conv6_1
NC_conv6_1, NC_conv2_2	Concat	–	256	–	$\frac{h}{2} \times \frac{w}{2}$	NC_concat2
NC_concat2	Upsampling	–	256	–	$h \times w$	NC_up3
NC_up3	Conv&ReLU	3×3	64	1	$h \times w$	NC_conv7_1
NC_conv7_1, NC_conv1_2	Concat	–	128	–	$h \times w$	NC_concat3
NC_concat3	Conv	3×3	3	–	$h \times w$	NC_conv8
NC_conv8	Tanh	–	3	–	$h \times w$	SDNC_residual
SDNC_residual, SDNU_noise	Addition	–	3	–	$h \times w$	SDNC_noise

Table 3. Architecture of the neighboring correlation network. Upsampling indicates the nearest interpolation operation. SDNU noise and SDNC noise are signal-dependent and neighboring uncorrelated noise and signal-dependent and neighboring correlated noise.

Inputs	Operator	Kernel	Output Channels	Stride	Output Size	Output Name
Noise	Conv	3×3	64	1	96×96	D1_conv1
D1_conv1	Conv&LReLU	4×4	64	2	48×48	D1_conv2
D1_conv2	Conv&LReLU	3×3	128	1	48×48	D1_conv3
D1_conv3	Conv&LReLU	4×4	128	2	24×24	D1_conv4
D1_conv4	Conv&LReLU	3×3	256	1	24×24	D1_conv5
D1_conv5	Conv&LReLU	4×4	256	2	12×12	D1_conv6
D1_conv6	Conv&LReLU	3×3	512	1	12×12	D1_conv7
D1_conv7	Conv&LReLU	4×4	512	2	6×6	D1_conv8
D1_conv8	Conv&LReLU	3×3	512	1	6×6	D1_conv9
D1_conv9	Conv&LReLU	4×4	512	2	3×3	D1_conv10
D1_conv10	Flatten	–	4608	–	1×1	D1_FC1
D1_FC1	Conv	1×1	100	1	1×1	D1_FC2
D1_FC2	Conv	1×1	1	1	1×1	D1_score

Table 4. Architecture of the discriminator D_1. Flatten refers to the process of transforming a three-dimensional tensor into a one-dimensional vector through a reshaping operation. Note that the size of input noise patch is 96×96.

Inputs	Operator	Kernel	Output Channels	Stride	Output Size	Output Name
Noise	Conv	3×3	64	1	32×32	D2_conv1
D2_conv1	Conv&LReLU	4×4	64	2	16×16	D2_conv2
D2_conv2	Conv&LReLU	3×3	128	1	16×16	D2_conv3
D2_conv3	Conv&LReLU	4×4	128	2	8×8	D2_conv4
D2_conv4	Conv&LReLU	3×3	256	1	8×8	D2_conv5
D2_conv5	Conv&LReLU	4×4	256	2	4×4	D2_conv6
D2_conv6	Flatten	–	4096	–	1×1	D2_FC1
D2_FC1	Conv	1×1	100	1	1×1	D2_FC2
D2_FC2	Conv	1×1	1	1	1×1	D2_score

Table 5. Architecture of the discriminator D_2. Note that the size of input noise patch is 32×32.

real noise $(\boldsymbol{n}) \downarrow_s$ is greatly attenuated. Therefore, we calculate the adversarial loss between the two down-sampled noise maps.

3. Discriminator Loss Functions

We adopt WGAN-GP [2] to reduce the discrepancies between the generated noise distribution and the real noise distribution. The adversarial loss $\mathcal{L}_{D_1}$ for the discriminator D_1 is defined as:

$$\begin{aligned}\mathcal{L}_{D_1} = &\mathbb{E}_{\hat{\boldsymbol{n}}}\left[D_1(\hat{\boldsymbol{n}})\right] - \mathbb{E}_{\boldsymbol{n}}\left[D_1(\boldsymbol{n})\right] \\ &+ \lambda_{gp}\,\mathbb{E}_{\hat{\boldsymbol{n}}_\delta}\left[(||\nabla_{\hat{\boldsymbol{n}}_\delta} D_1(\hat{\boldsymbol{n}}_\delta)||_2 - 1)^2\right],\end{aligned} \tag{1}$$

where $\hat{\boldsymbol{n}}_\delta$ is the interpolated noise by computing as a weighted combination of the generated SDNC noise $\hat{\boldsymbol{n}}$ and real noise $\boldsymbol{n}$, and λ_{gp} is the weight for the gradient penalty term [2], which is set to 10 in this paper. Similarly, the adversarial loss for the discriminator D_2 is described as:

$$\begin{aligned}\mathcal{L}_{D_2} = &\mathbb{E}_{(\hat{\boldsymbol{v}})\downarrow_s}\left[D_2((\hat{\boldsymbol{v}})\downarrow_s)\right] - \mathbb{E}_{(\boldsymbol{n})\downarrow_s}\left[D_2((\boldsymbol{n})\downarrow_s)\right] \\ &+ \lambda_{gp}\,\mathbb{E}_{\hat{\boldsymbol{v}}_\delta}\left[(||\nabla_{\hat{\boldsymbol{v}}_\delta} D_2(\hat{\boldsymbol{v}}_\delta)||_2 - 1)^2\right],\end{aligned} \tag{2}$$

where $\hat{\boldsymbol{v}}_\delta$ is also a weighted combination of two down-sampled noise $(\hat{\boldsymbol{v}})\downarrow_s$ and $(\boldsymbol{n})\downarrow_s$, similar to the interpolated noise $\hat{\boldsymbol{n}}_\delta$.

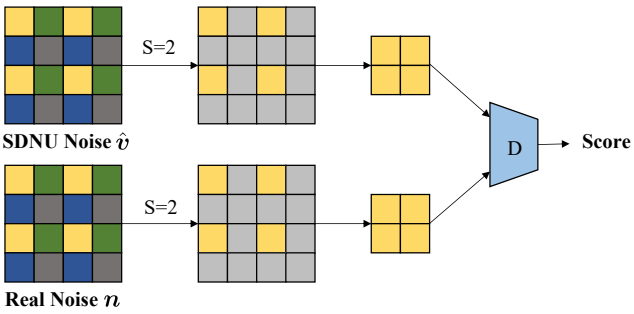

Figure 1. Illustration of Pixel-Shuffle Down-Sampling (PD) scheme. The down-sampled stride in this example is 2. SDNU noise is the signal-dependent and neighboring uncorrelated noise. D denotes the discriminator.

4. Network Effectiveness Evaluation

In the first case, for each noisy-clean image pair from the training set, GENet estimates the gain factor from the noisy image, which is used as the noise level to synthesize AWGN on the corresponding clean image. In the second case, GENet and NPNet are applied to synthesize signal-dependent and neighboring uncorrelated (SDNU) noise on clean images. In the third case, we generate signal-independent and neighboring correlated (SINC) noise on clean images by using GENet and NCNet. In the last case, we synthesize signal-dependent and neighboring correlated (SDNC) noise by using the whole framework. We finally train the DnCNN on the training set, where the noisy images are either the synthesized or real ones and evaluate the denoising performance of DnCNN on the validation set. Table 6 shows the denoising results of denoisers trained on different types of synthetic noise. We observe DnCNN trained on the SDNC noise performs best, suggesting modeling both

GENet	NPNet	NCNet	PSNR
✓			36.29
✓	✓		37.10
✓		✓	39.37
✓	✓	✓	**39.46**

Table 6. Ablation study of different networks. The corresponding types of synthetic noise from the top to bottom are AWGN; signal-dependent and neighboring uncorrelated (SDNU) noise; signal-independent and neighboring correlated (SINC) noise; and signal-dependent and neighboring correlated (SDNC) noise.

signal dependency and neighboring correlation of noise is crucial for sRGB real noise synthesis. Moreover, the denoising performance of DnCNN in the third case outperforms its counterpart in the second case, indicating that explicitly modeling the neighboring correlation of sRGB real noise enables the proposed noise synthesizing framework to greatly narrows the gap between the synthetic noise and sRGB real noise.

References

[1] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016. 1

[2] Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron C Courville. Improved training of wasserstein gans. *NIPS*, 2017. 2

[3] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 1

[4] Min Lin, Qiang Chen, and Shuicheng Yan. Network in network. *arXiv preprint arXiv:1312.4400*, 2013. 1

[5] Andrew L Maas, Awni Y Hannun, Andrew Y Ng, et al. Rectifier nonlinearities improve neural network acoustic models. In *ICML*, 2013. 1

[6] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *MICCAI*, 2015. 1

[7] Yuqian Zhou, Jianbo Jiao, Haibin Huang, Yang Wang, Jue Wang, Honghui Shi, and Thomas Huang. When awgn-based denoiser meets real noises. In *AAAI*, 2020. 1