Supplementary Material
Cross-Patch Graph Convolutional Network for Image Denoising

Yao Li, Xueyang Fu, Zheng-Jun Zha
University of Science and Technology of China
xslx@mail.ustc.edu.cn, {xyfu, zhazj}@ustc.edu.cn

In this supplementary material, we first introduce our data augmentation model. Then, we perform noise analysis to show that our synthetic noise distribution is very close to the real noise distribution in both the raw images and the sRGB images. Finally, the effectiveness of DC-Net is verified.

1. Noisy Image Generation Model

In this section, we describe the details of our method to synthesize realistic noisy sRGB images. The proposed procedure is shown in Fig. 1.

1.1. Inverse ISP

Even if the characteristic of noise in sRGB space is unknown, the noise of raw data is well understood. We transfer an image from the sRGB space to the raw space for noise addition. Some methods have discussed the ISP pipeline, such as [2, 8, 13]. Inspired by these works, we propose a simplified but effective inverse ISP procedure, including the most common ISP components. The ISP pipeline consists of white balancing, demosaicing, color space conversion, gamma transform, and tone mapping. In the inverse ISP pipeline, the input is an sRGB image and the output is a simulated raw image. We denote $\mathbf{x}$ as the input and $\mathbf{y}$ as the output in every inverse ISP component for simplicity.

1.1.1 Inverse Tone Mapping

The S-shaped curve is usually used to obtain better visual perception on an image, which is called tone mapping. Our simulated tone mapping function and its inverse are defined as:

$$\mathbf{x} = 0.5 - 0.5\cos(\pi \mathbf{y}), \qquad \mathbf{y} = \frac{1}{\pi}\cos^{-1}(1 - 2\mathbf{x}). \quad (1)$$

1.1.2 Inverse Gamma transform

Gamma transform is the most widely used non-linear processing in ISP, which has the form of $\mathbf{y} = \mathbf{x}^{\lambda}$. We adopt $\lambda = 2.2$ as most platforms such as Windows and Mac.

1.1.3 Inverse Color Space Conversion

In order to make an image have the same visual effect as possible on various display devices, the demosaiced image is needed to be transformed to the device-independent color space XYZ and then to the sRGB space. These two transforms are denoted by $M_{\mathrm{Raw}\to\mathrm{XYZ}}$ and $M_{\mathrm{XYZ}\to\mathrm{sRGB}}$. For inverting this color space conversion, we have:

$$\begin{bmatrix} \mathbf{y_r} \\ \mathbf{y_g} \\ \mathbf{y_b} \end{bmatrix} = \left(M^{-1}_{\mathrm{Raw}\to\mathrm{XYZ}} \cdot M^{-1}_{\mathrm{XYZ}\to\mathrm{sRGB}}\right) \begin{bmatrix} \mathbf{x_r} \\ \mathbf{x_g} \\ \mathbf{x_b} \end{bmatrix}. \quad (2)$$

From the metadata of an image, the transform matrix $M_{\mathrm{Raw}\to\mathrm{XYZ}}$ can be derived. As for $M_{\mathrm{XYZ}\to\mathrm{sRGB}}$, it is a standard proposed by CIE (Commission Internationale de L'Eclairage) in 1931.

1.1.4 Inverse Demosaicing

The original raw data captured by a single sensor with the Bayer filter has red, green, and blue three primary color components. To convert an image from the Bayer pattern to an sRGB image, we need to interpolate the two missing color values in each pixel. For simplicity, DND [12] and the method in [2] use the bi-linear interpolation, and SIDD [1] uses another simple linear interpolation [9]. In practical application, cameras usually employ more complicated interpolation methods like interpolation based on chromatic aberration [4], non-local equilibrium, and interpolation based on gradient. Although the interpolation method is unknown for an image, these methods are based on an assumption that the pixel color values at the original Bayer pattern positions are accurate and unchanged[1]. Thanks to this property, we sample the original pixel values in the Bayer pattern positions to recover the original Bayer pattern based on the metadata.

[1]For example, the red component at a position of a Bayer pattern keeps unchanged, while the green and blue components at this position are interpolated.

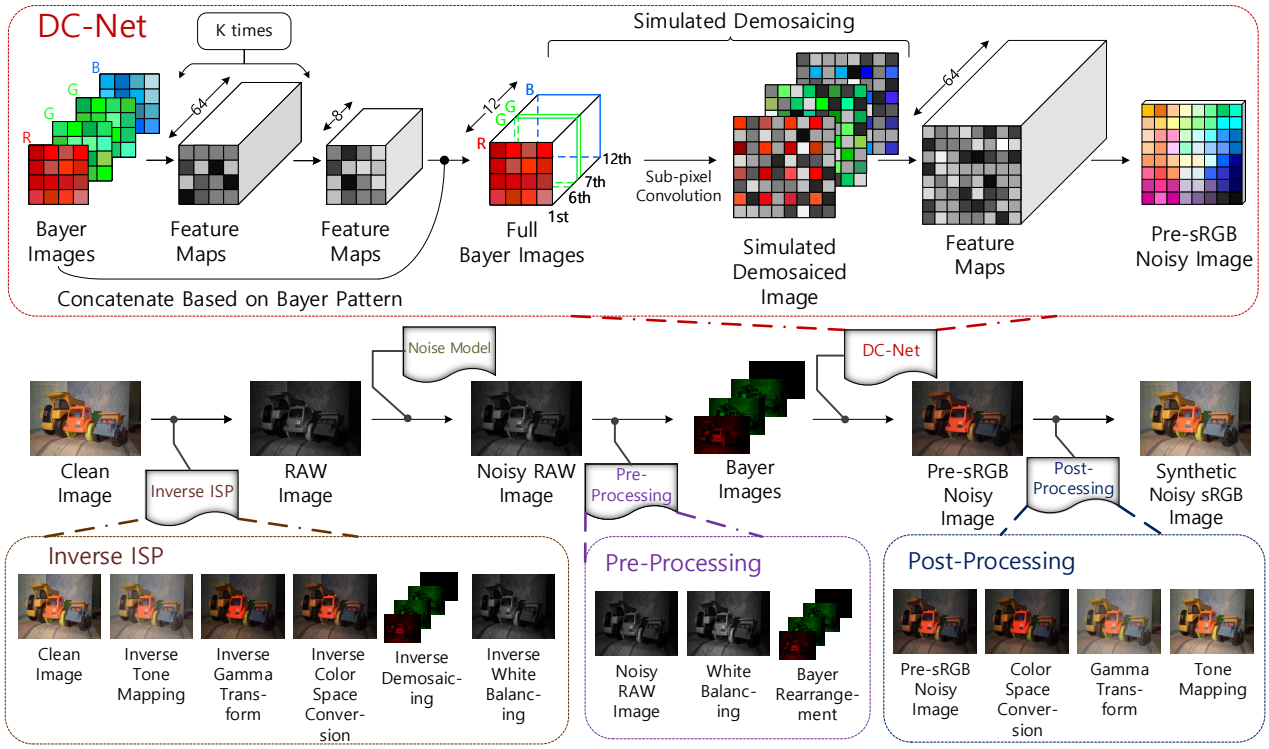


Figure 1. A flowchart illustrating the main steps of realistic noisy image generation in our procedure. A clean sRGB image is passed through the inverse ISP module to get a raw image, and after been added noise, the raw image is pre-processed to obtain the input to the DC-Net, which generates a pre-sRGB noisy image. Finally, the realistic noisy sRGB image is obtained by the post-processing.

1.1.5 Inverse White Balancing

White balance in digital photography is to adjust the colors of an image so that the image looks more natural. We go through the process of adjusting colors to primarily get rid of color casts, in order to make the image match what we see when we take it, since most light sources (the sun, light bulbs, flashlights, etc.) do not emit purely white color and have a certain color temperature. In practice, the three color values R, G, and B in the raw data are multiplied by different gains to correct color casts, and then the white and black level clipping is used to keep the pixel values in the range of $[0, 1]$. The gains are stored in the metadata. In inverse white balancing, we adopt the same pipeline as [2] to simulate the inverse white and black level clipping.

1.2. Noise Model

The noise in the raw data can be divided into two categories: shot noise and read noise. Shot noise is caused by the random arrival of photons. This is a fundamental trait of light. Since each photon is an independent event, the arrival of any given photon cannot be precisely predicted; instead, the probability of its arrival in a given period of time is governed by a Poisson distribution. With enough samples, a graph plotting the arrival of photons shows the familiar bell curve with a long tail on the right.

Read noise, which is also called electronic noise, is an uncertainty introduced by electrons when the electrons are converted to digital signals via the preamplifiers and the analog-to-digital converters (ADC). The read noise exhibits a Gaussian distribution with zero mean and fixed variance both in CMOS and CCD sensors [5].

The read noise is related to imperfections in the sensor electronics and is independent of the intensity of the light hitting the sensor. The independence between the read noise and shot noise means the variance of their sum is equal to the sum of their variances.

The intensities of these two kinds of noise are different under different brightness. The shot noise is dominant in the bright areas, while the read noise dominates in the dark [7]. In the dark areas, the distribution of the noise obeys Gaussian distribution. In brighter areas, the shot noise dominates with a large Poisson distribution parameter λ. We assume X_λ denotes the noisy intensities/colors of images and obeys Poisson distribution:

$$P_x = \frac{\lambda^x e^{-\lambda}}{x!}. \tag{3}$$

The moment generating function of X_λ is:

$$\lim_{\lambda \to \infty} M_{X_{\lambda(t)}} = E\left[e^{tX_\lambda}\right] = e^{\lambda\left(e^t - 1\right)}. \tag{4}$$

Then consider a standardized Poisson random variable $\frac{X_\lambda - \lambda}{\sqrt{\lambda}}$. We calculate its moment generating function:

$$\begin{aligned}
\lim_{\lambda\to\infty} M_{X_{\lambda(t)}} &= \lim_{\lambda\to\infty} E\left[\exp\left(\frac{X_\lambda-\lambda}{\sqrt{\lambda}}t\right)\right] \\
&= \lim_{\lambda\to\infty} \exp\left(-t\sqrt{\lambda}\right) E\left[\exp\left(\frac{tX_\lambda}{\sqrt{\lambda}}\right)\right] \\
&= \lim_{\lambda\to\infty} \exp\left(-t\sqrt{\lambda}\right) \exp\left(\lambda\left(e^{\frac{t}{\sqrt{\lambda}}-1}\right)\right) \\
&= \lim_{\lambda\to\infty} \exp\left(-t\sqrt{\lambda}+\lambda\left(\frac{t}{\sqrt{\lambda}}+\frac{t^2}{2\lambda}+\cdots\right)\right) \\
&= \lim_{\lambda\to\infty} \exp\left(\frac{t^2}{2}+\frac{t^3}{6\sqrt{\lambda}}+\cdots\right) \\
&= \exp\left(\frac{t^2}{2}\right).
\end{aligned} \tag{5}$$

This implies that the associated unstandardized random variable X_λ has a limiting distribution that is Gaussian with mean λ and variance λ. For large enough λ, this distribution can be regarded as a Gaussian distribution. The approximation is:

$$\mathcal{P}(\lambda) \approx \mathcal{N}(\lambda, \lambda). \tag{6}$$

Because of the additivity of Gaussian distribution, we construct a realistic noise model of the raw image as [5]:

$$\mathbf{y} = \mathbf{x} + \mathcal{N}(\mathbf{0}, \sigma(\mathbf{x})), \qquad \sigma^2(\mathbf{x}) = \beta_s \mathbf{x} + \beta_r, \tag{7}$$

where β_s is the signal-dependent component of the shot noise, and β_r is the independent component of the read noise. $\mathbf{x}$ and $\mathbf{y}$ represent the clean and noisy raw images, respectively.

With greater levels of luminosity, lower ISO, and longer exposure time, the image becomes cleaner with less noise. However, the read noise does not disappear but is closer to a threshold since it is a kind of sensor-dependent noise. We validate our observation by calculating the noise parameters of the images in the SIDD data set. The result is shown in Fig. 2. Based on the statistics, we use an exponential function to estimate the noise level functions (NLFs):

$$\ln(\beta_r) = \mathrm{a}e^{\mathrm{b}\ln(\beta_s)} - \mathrm{c} + \mathcal{N}(0, \sigma). \tag{8}$$

We use the Levenberg-Marquardt algorithm [10] to estimate the parameters of Eq. 8, and obtain $\mathrm{a} = 71.28, \mathrm{b} = 0.5496$, and $\mathrm{c} = 13.86$. Empirically, we set $\sigma = 0.3$. In our experiment, we randomly select $\ln(\beta_s)$ from the range of $[-9.1, -3.8]$ to generate the shot noise of each image. Then β_r is derived from Eq. 8 to generate the read noise.

1.3. Pre-Prosessing

In order to restore the sRGB image from the noisy raw image, the ISP is carried out. Some ISP components are invertible but others are not. Usually, in a simplified ISP pipeline, all the ISP components are invertible except demosaicing and white and black level clipping. We design a network to learn Demosaicing and to reduce the loss by the Clipping, which is named DC-Net. Before the network, we pre-process the raw image to relieve the training burden of DC-Net, due to the following reasons. Firstly, it is of great difficulty in directly learning a pipeline from raw images to sRGB images, which is validated by our experiment. So it is better for DC-Net to focus on specific stages in ISP. Secondly, it is better to design a network to learn the unknown demosaicing rather than to apply a simple bi-linear interpolation. To begin with, we perform white balancing on the raw image and apply the black and white level clipping[2]. Then we rearrange the Bayer pattern to a fixed format (e.g., RGGB) as the input of DC-Net (see Fig. 2). As a result, we simplify the problem of DC-Net's training to mainly learn both demosaicing and how to reduce the loss generated by the clipping. The output of DC-Net is called the pre-sRGB noisy image.

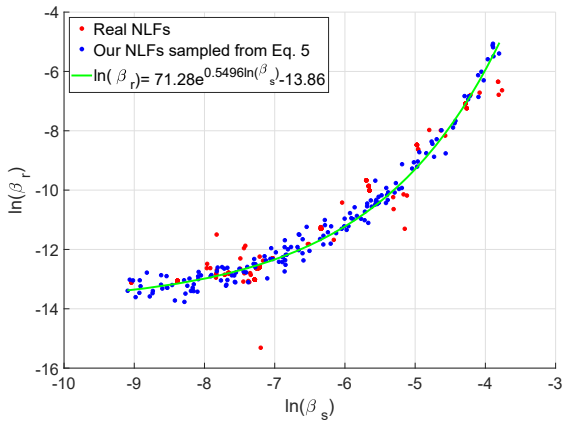


Figure 2. Comparison between the real NLFs and our NLFs. The real NLFs are recorded in the metadata of cameras. Our NLFs are randomly sampled from Eq. 8.

1.4. DC-Net

In inverse demosaicing, only the pixel values in the positions of the original Bayer pattern are sampled and the interpolated pixels are abandoned. As a result, it is impossible to restore its demosaiced image with only the raw noisy image. Some works simplify this procedure as bi-linear interpolation such as [11, 2]. Nevertheless, it is artifact-prone and the most naive demosaicing technique. Besides, the whole ISP is a simulated process with errors. For example, the black and white level clipping is irreversible which is adopted after white balancing and color space conversion. DC-Net is

[2]The clipping is necessary because after white balancing, some pixel values are out of the range $[0, 1]$.

designed to tackle these problems.

Our DC-Net is inspired by [6], but with two improvements. First, in the demosaicing network [6], we find that the artifacts on the high-frequency regions mainly come from the concatenated masks. Therefore, in our DC-Net, we use the unmasked original input for the concatenation. Second, a new module called simulated demosaicing is added. Based on the property of demosaicing where the original pixel colors in the Bayer pattern are unchanged, this module can explicitly force DC-Net to keep these pixel colors.

As shown in DC-Net in Fig. 1, we first send the rearranged Bayer images (e.g., RGGB) from the Bayer pattern into K convolutional layers to learn the relationship among the pixels. All the channels of the feature maps are set to 64. Then in the (K+1)th convolutional layer, we reduce the number of filters to 8. We use the rectified linear unit (ReLU) as the activation function. All the filters are 3×3. After that, the input Bayer images and the 8 feature maps are concatenated. Note that these Bayer images (RGGB) are inserted at the 1st, 6th, 7th, and 12th locations so that the following sub-pixel convolution [14] can directly generate three demosaiced R, G, and B channels. Finally, a convolutional layer with 64 3×3 filters generates 64 feature maps, which are fed into the last convolutional layer with 3 1×1 filters to obtain the pre-sRGB image. The following mean square error loss is used to train the network:

$$L = \frac{1}{N} \sum_{i=1}^{N} \|\hat{\mathbf{y}}_i - \mathbf{y}_i\|_2^2 , \tag{9}$$

where $\mathbf{y}_i$ is the output of DC-Net, and $\hat{\mathbf{y}}_i$ as the ground truth is obtained through the processing of the *i*th sRGB image by three inverse ISP components: inverse tone mapping, inverse Gamma transform, and inverse color space conversion. Note that for the training of DC-Net, we do not add additional noise with our noise model to the raw image; instead, we train it using the real-world noisy images

1.5. Post-Processing

The pre-sRGB image finally goes through post-processing before being viewed, which in order consists of color space conversion, gamma transform, and tone mapping. Theoretically, owing to the invertibility of these post-processing components, the original image can be well restored as long as the output of DC-Net is real.

2. Noise Analysis

The key to generate realistic noisy images is that the generated raw images have noise properties similar to those of real raw images. We evaluate it in terms of both the NLFs of the raw images and the noise levels of the sRGB images.

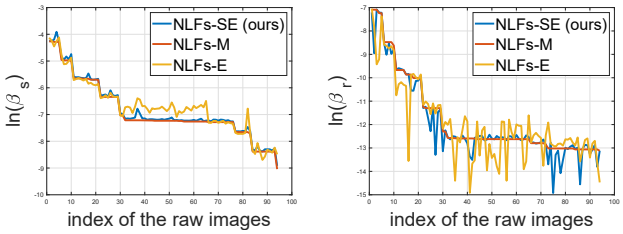


Figure 3. Statistics of the estimated NLFs. Our NLFs-SE curves for both β_s and β_r are close to NLFs-E and NLFs-M, indicating the similarity between our raw noise and the real raw noise.

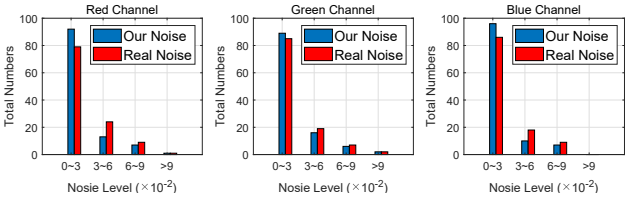


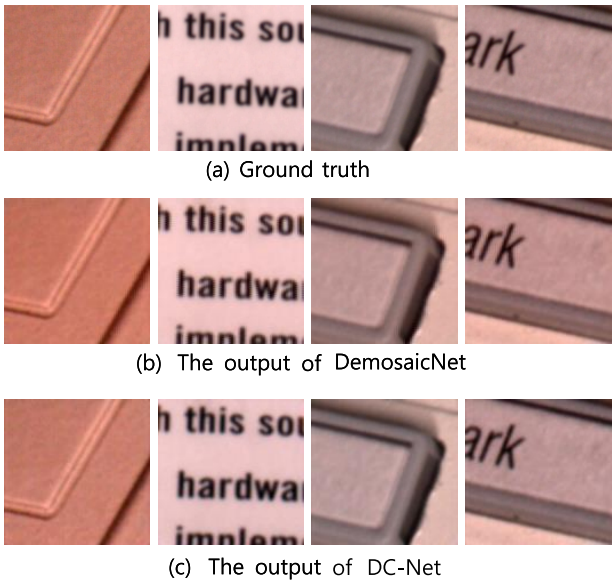
Figure 4. Noise levels of the sRGB images. We estimate the noise levels in the real noisy images and our synthetic noisy images. The distributions of our noise are similar to the real ones in all the three channels (the pixel values are normalized to [0, 1]).

2.1. NLFs of the Raw Images

Noise in the raw space is determined by shot and read noise. It is an effective way to evaluate the synthetic noise by estimating the NLFs of the generated raw images. We calculate the Poisson-Gaussian noise parameters β_s and β_r on SIDD by using [5], which is an effective method to estimate the NLFs. For a noisy and noise-free raw image pair in SIDD, the metadata of the noisy image raw image records its NLF, which is denoted as NLF-M. The NLF of this raw image can also be estimated with [5], denoted as NLF-E. We apply the proposed inverse ISP and noise model to the corresponding noise-free sRGB image, generating the simulated noisy raw image, of which the NLF is again estimated with [5], denoted as NLF-SE. The comparison of the three NLFs is given in Fig. 3, where the three curves are obtained from the raw noisy images. We can see that overall our NLFs-SE curves for β_s and β_r are close enough to the NLFs-M and NLFs-E, showing that the noise properties of our simulated noisy raw images are realistic enough.

2.2. Noise Levels of the sRGB Images

To figure out whether our noise is close to the real noise in the sRGB noisy images, we use the noise level estimation method proposed by [3]. We calculate the noise levels of SIDD noisy images as well as our synthetic noisy sRGB images for each channel. All of our images take the same NLFs as the real noisy images for a fair comparison. We draw the histograms based on the results, which are shown in Fig. 4. We can see that the distributions of our noise are similar to the real ones on all three channels. The major-



(a) Ground truth

(b) The output of DemosaicNet

(c) The output of DC-Net

Figure 5. Visual demosaicing comparison. (a) Ground truth. (b) The output of DemosicNet. (c) The output of DC-Net. DemosaicNet results in the zigzag effect on the edges while DC-Net preserves the sharpness of them (best viewed on screen with zooming in).

Table 1. Denoising results of DnCNN trained by the synthetic noisy images generated with different demosaicing methods on the SIDD testing set.

	Bi-linear	DemosaicNet	DC-Net
PSNR	30.03	35.13	**35.35**
SSIM	0.5038	0.8106	**0.8325**

ity of the noise levels are below 0.03. All pixel values are normalized to [0, 1].

3. Effectiveness of DC-Net

The goal of DC-Net is to convert the Bayer pattern to the pre-sRGB image, which is similar to other demosaicing methods. We compare it with DemosaicNet [6] which is one of the state-of-the-arts. We train both DemosaicNet and DC-Net using the real-world noisy images from the SIDD training set for demosaicing. After training, each of them is used in our ISP pipeline to generate noisy sRGB images, which are then employed to train DnCNN[15] for denoising. Their performances for denoising are compared in Table 1, which shows that DC-Net performs better. In fact, DC-Net also outperforms DemosaicNet for demosaicing, with PSNR 53.32dB vs. 53.16dB on the SIDD testing set. Fig. 5 shows a visual comparison. DemosaicNet simply upsamples the Bayer images to the original RGB image size, inserts zeros at the interpolation positions as a mask, and then the masked Bayer images are concatenated with the feature maps in the last two convolutional layers. This approach leads to the difficulty in filling the zeros with proper values and results in the zigzag effect on edges. Our approach exploits the nature of the Bayer images and reduces the distortion.

References

[1] Abdelrahman Abdelhamed, Stephen Lin, and Michael S. Brown. A high-quality denoising dataset for smartphone cameras. In *CVPR*, 2018.

[2] Tim Brooks, Ben Mildenhall, Tianfan Xue, Jiawen Chen, Dillon Sharlet, and Jonathan T. Barron. Unprocessing images for learned raw denoising. In *CVPR*, 2019.

[3] Guangyong Chen, Fengyuan Zhu, and Pheng-Ann Heng. An efficient statistical method for image noise level estimation. In *ICCV*, 2015.

[4] David R Cok. Signal processing method and apparatus for producing interpolated chrominance values in a sampled color image signal. In *US.Patent*, 1987.

[5] Alessandro Foi, Mejdi Trimeche, Vladimir Katkovnik, and Karen O. Egiazarian. Practical poissonian-gaussian noise modeling and fitting for single-image raw-data. *IEEE Trans. Image Processing*, 17(10):1737–1754, 2008.

[6] Michaël Gharbi, Gaurav Chaurasia, Sylvain Paris, and Frédo Durand. Deep joint demosaicking and denoising. *ACM Trans. Graph.*, 35(6):191:1–191:12, 2016.

[7] Heli T Hytti. Characterization of digital image noise properties based on raw data. In *Image Quality and System Performance III*, volume 6059, page 60590A. International Society for Optics and Photonics, 2006.

[8] Zhetong Liang, Jianrui Cai, Zisheng Cao, and Lei Zhang. Cameranet: A two-stage framework for effective camera isp learning. *arXiv preprint arXiv:1908.01481*, 2019.

[9] Henrique S. Malvar, Li-wei He, and Ross Cutler. High-quality linear interpolation for demosaicing of bayer-patterned color images. In *ICASSP*, 2004.

[10] Jorge J Moré. The levenberg-marquardt algorithm: implementation and theory. In *Numerical Analysis*, pages 105–116. Springer, 1978.

[11] Tobias Plotz and Stefan Roth. Benchmarking denoising algorithms with real photographs. In *CVPR*, 2017.

[12] Tobias Plötz and Stefan Roth. Neural nearest neighbors networks. In *NeurIPS*, 2018.

[13] Nguyen Ho Man Rang and Michael S. Brown. RAW image reconstruction using a self-contained srgb-jpeg image with only 64 KB overhead. In *CVPR*, 2016.

[14] Wenzhe Shi, Jose Caballero, Ferenc Huszar, Johannes Totz, Andrew P. Aitken, Rob Bishop, Daniel Rueckert, and Zehan Wang. Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network. In *CVPR*, 2016.

[15] Kai Zhang, Wangmeng Zuo, Yunjin Chen, Deyu Meng, and Lei Zhang. Beyond a gaussian denoiser: Residual learning of deep CNN for image denoising. *IEEE Trans. Image Processing*, 26(7):3142–3155, 2017.