

Learning Deep Convolutional Networks for Demosaicing

Nai-Sheng Syu*, Yu-Sheng Chen*, Yung-Yu Chuang

arXiv:1802.03769v1 [cs.CV] 11 Feb 2018

Abstract—This paper presents a comprehensive study of applying the convolutional neural network (CNN) to solving the demosaicing problem. The paper presents two CNN models that learn end-to-end mappings between the mosaic samples and the original image patches with full information. In the case the Bayer color filter array (CFA) is used, an evaluation on popular benchmarks confirms that the data-driven, automatically learned features by the CNN models are very effective and our best proposed CNN model outperforms the current state-of-the-art algorithms. Experiments show that the proposed CNN models can perform equally well in both the sRGB space and the linear space. It is also demonstrated that the CNN model can perform joint denoising and demosaicing. The CNN model is very flexible and can be easily adopted for demosaicing with any CFA design. We train CNN models for demosaicing with three different CFAs and obtain better results than existing methods. With the great flexibility to be coupled with any CFA, we present the first data-driven joint optimization of the CFA design and the demosaicing method using CNN. Experiments show that the combination of the automatically discovered CFA pattern and the automatically devised demosaicing method outperforms other patterns and demosaicing methods. Visual comparisons confirm that the proposed methods reduce more visual artifacts. Finally, we show that the CNN model is also effective for the more general demosaicing problem with spatially varying exposure and color and can be used for taking images of higher dynamic ranges with a single shot. The proposed models and the thorough experiments together demonstrate that CNN is an effective and versatile tool for solving the demosaicing problem.

Index Terms—Convolutional neural network, demosaicing, color filter array (CFA).

I. INTRODUCTION

MOST digital cameras contain sensor arrays covered by color filter arrays (CFAs), mosaics of tiny color filters. Each pixel sensor therefore only records partial spectral information about the corresponding pixel. Demosaicing, a process of inferring the missing information for each pixel, plays an important role to reconstruct high-quality full-color images [2], [3], [4]. Since demosaicing involves prediction of missing information, there are inevitably errors, leading to visual artifacts in the reconstructed image. Common artifacts include the zipper effects and the false color artifacts. The former refers to abrupt or unnatural changes of intensities over neighboring pixels while the later is for the spurious colors that are not present in original image. In principle, the CFA design and the demosaicing method should be devised jointly

*Nai-Sheng Syu and Yu-Sheng Chen contributed equally to this work. This work is based on Nai-Sheng Syu's master thesis [1]. All authors are with the Department of Computer Science and Information Engineering, National Taiwan University, Taipei, Taiwan, 106. E-mail: vm3465s939873 | nothinglo | cyy@cmlab.csie.ntu.edu.tw.

for reducing visual artifacts as much as possible. However, most researches only focus on one of them.

The Bayer filter is the most popular CFA [5] and has been widely used in both academic researches and real camera manufacturing. It samples the green channel with a quincunx grid while sampling red and blue channels by a rectangular grid. The higher sampling rate for the green component is considered consistent with the human visual system. Most demosaicing algorithms are designed specifically for the Bayer CFA. They can be roughly divided into two groups, interpolation-based methods [6], [7], [8], [9], [10], [11], [12], [13], [14] and dictionary-based methods [15], [16]. The interpolation-based methods usually adopt observations of local properties and exploit the correlation among wavelengths. However, the handcrafted features extracted by observations have limitations and often fail to reconstruct complicated structures. Although iterative and adaptive schemes could improve demosaicing results, they have limitations and introduce more computational overhead. Dictionary-based approaches treat demosaicing as a problem of reconstructing patches from a dictionary of learned base patches. Since the dictionary is learned, it can represent the distribution of local image patches more faithfully and provide better color fidelity of the reconstructed images. However, the online optimization for reconstruction often takes much longer time, making such methods less practical.

Despite its practical use for decades, researches showed the Bayer CFA has poor properties in the frequency-domain analysis [17]. Thus, some efforts have been put in proposing better CFA designs for improving color fidelity of the demosaiced images [18], [19], [20]. Earlier work mainly focused on altering the arrangement of RGB elements to get better demosaicing results in terms of some handcrafted criteria. Some also explored color filters other than primary colors. Inspired by the frequency representation of mosaiced images [17], several theoretically grounded CFA designs have been proposed [18], [19]. They however involve considerable human effort. Recently, automatic methods for generating CFAs have been proposed by exploiting the frequency structure, a matrix recording all the luminance and chrominance components of given mosaiced images [20]. However, although theoretically better, most of these CFAs can only reach similar performances as the state-of-the-art demosaicing methods with the Bayer CFA. The main reason is that the more complicated CFA designs require effective demosaicing methods to fully release their potential. Unfortunately, due to the complex designs, such demosaicing methods are more difficult to devise and, compared with the Bayer CFA, less efforts have been put into developing demosaicing methods for these CFAs.

We address these issues by exploring the convolutional neural network (CNN). Because of breakthroughs in theory and improvements on hardware, recently CNNs have shown promises for solving many problems, such as visual recognition, image enhancement and game playing. By learning through data, the network automatically learns appropriate features for the target applications. We first address the demosaicing problem with the popular Bayer CFA (Section III). Inspired by CNN models for super-resolution [21], [22], we present two CNN models, DeMosaicing Convolutional Neural Network (DMCNN, Section III-A) and Very Deep DMCNN (DMCNN-VD, Section III-B), for demosaicing. In contrast with hand-crafted features/rules by many interpolation-based methods, the CNN models automatically extract useful features and captures high-level relationships among samples. Experiments show that the CNN-based methods outperforms the state-of-the-art methods in both the sRGB space (Section III-C) and the linear space (Section III-D). In addition, they could perform denoising and demosaicing simultaneously if providing proper training data. We next show that the CNN-based methods can be easily adopted for demosaicing with CFA designs other than the Bayer one (Section IV). The data-driven optimization approach makes it easy to train the CNN-based demosaicing methods with different CFAs and outperform existing methods (Section IV-A). With its flexibility to be used with any CFA, we present the first data-driven method for joint optimization of the CFA design and the demosaicing method (Section IV-B). Finally, we demonstrate that the CNN-based method can also be applied to solving a more challenging demosaicing problem where the filter array has spatially varying exposure and color (Section IV-C). It enables taking images with a higher dynamic range using a shingle shot. All together, the paper presents a comprehensive study which thoroughly explores the applications of CNN models to the demosaicing problems.

II. RELATED WORK

Color demosaicing. The demosaicing methods can be roughly classified into two categories: *interpolation-based* [6], [7], [8], [9], [10], [11], [12], [13], [14] and *dictionary-based* methods [15], [16]. Surveys of early methods can be found in some comprehensive review papers [2], [3], [4]. Recently, Kiku et al. [8] proposed a novel way to demosaic images in the residual space, and later extended the method to minimize Laplacian of the residual, instead of the residual itself [9]. The residual space is considered smoother and easier for reconstruction. Monno et al. [10] proposed an iterative, adaptive version of residual interpolation framework.

CFA design. Alleysson et al. [17] analyzed the demosaicing problem in the frequency domain. In the frequency domain, the CFA pattern is usually decomposed into three components, luminance and two chrominance frequencies. Hirakawa et al. [18] formulated CFA design as a problem to maximize the distances between luminance and chrominance components and obtained the optimal pattern by exhaustive search in the parameter space. Condat [23] followed the same spirit and proposed a CFA design that is more robust to noise, aliasing, and low-light circumstances. Hao et al. [19] and

Bai et al. [20] each introduced a pattern design algorithm based on the frequency structure proposed by Li et al. [24]. Hao et al. [19] formulated the CFA design problem as a constrained optimization problem and solved it with a geometric method. Later, Bai et al. [20] introduced an automatic pattern design process by utilizing a multi-objective optimization approach which first proposes frequency structure candidates and then optimizes parameters for each candidate.

General demosaicing. In addition to colors, other properties of light, such as exposures (spatially varying exposure, SVE) and polarization, could also be embedded into the filter array and more general demosaicing algorithms can be used for recovering the missing information. Nayar et al. [25] proposed a general demosaicing framework, *Assorted Pixel*, by assuming the demosaiced result can be obtained from an n -degree polynomial function of neighboring mosaiced pixels. The whole process can therefore be thought as a regression problem by solving a linear system. Yasuma et al. [26] later proposed a more general pattern, *Generalized Assorted Pixel*, with the capability to recover monochrome, RGB, high dynamic range (HDR), multi-spectral images while sacrificing spatial resolutions. We adopt a similar spatially varying exposure and color (SVEC) setting as Nayar et al. [25] to demonstrate the potential of the CNN-based methods for generalized demosaicing.

Convolution neural networks. To date, deep learning based approaches have dominated many high-level and low-level vision problems. Krizhevsky et al. [27] showed the deep CNN is very effective for the object classification problem. In addition to high-level vision problems, CNN is also found effective in many low-level image processing problems, including deblurring [28], [29], denoising [30], [31], super resolution [21], [22], [32], [33], colorization [34], photo adjustment [35] and compression artifacts reduction [36]. Inspired by the successful CNN-based super-resolution methos [21], [22], this paper attempts to address both the demosaicing problem and the CFA design problem using end-to-end CNN models.

There were few attempts on applying CNN models to solving the demosaicing problem [1], [37], [38]. In SIGGRAPH Asia 2016, Gharbi et al. [37] proposed a CNN model for joint demosaicing and denoising. It downsamples the mosaic image into a lower-resolution feature map and uses a series of convolutions for computing the residual at the lower resolution. The input mosaic image is then concatenated with the upsampled residual. The final output is constructed by a last group of convolutions at the full resolution and then a linear combination of resultant feature maps. In ICME 2017, Tan et al. [38] proposed a CNN model for Bayer demosaicing. The model first uses bilinear interpolation for generating the initial image and then throws away the input mosaic image. Given the initial image as the input, the model has two stages for demosaicing. The first stage estimates G and R/B channels separately while the second stage estimates three channels jointly. Both papers only tackled the Bayer demosaicing problem. On the other hand, this paper addresses a wider set of demosaicing problems, including demosaicing in the linear space, demosaicing with non-Bayer patterns, CNN-based pattern design and demosaicing with a SVEC pattern.

III. DEMOSAICING WITH THE BAYER FILTER

The Bayer filter is the most popular CFA. In this section, we will focus on demosaicing with the Bayer filter using the convolutional neural network. First, we will introduce two CNN architectures, DMCNN (Section III-A) and DMCNN-VD (Section III-B), respectively inspired by recent successful CNN models for image super-resolution, SRCNN [21] and VDSR [22].

A. Demosaicing convolutional neural network (DMCNN)

The architecture of the demosaicing convolutional neural network (DMCNN) is inspired by Dong et al.'s SRCNN [21] for super-resolution. Fig. 1 gives the architecture of DMCNN. Since relevant information for demosaicing is often only present locally, patches are densely extracted and presented as the inputs to the network. We used 33×33 patches for DMCNN. Each pixel of the patch consists of three color channels by leaving the two missing channels blank. Thus, the input size is $33 \times 33 \times 3$. Another option would be to have $33 \times 33 \times 1$ patches by using the mosaiced image patch directly. It however could be difficult for the network to figure out which color channel each pixel represents. Four separate networks could be necessary for the four different locations in the Bayer pattern. We found that it is more effective to simply leave missing channel as blank and learn a unified network for different locations of the CFA tile. This way, the designed network is also more flexible for different CFA patterns as we will explore in Section IV. Similar to SRCNN, DMCNN consists of three layers, each for a specific task:

- 1) **Feature extraction layer.** The first layer is responsible for extracting useful local features. We use 128 9×9 filters which are initialized as Gaussian kernels. The output of this layer can be regarded as a low-resolution map of 128-d feature vectors.
- 2) **Non-linear mapping layer.** The function of the second layer is to map the extracted high-dimension feature vectors to lower-dimension ones. We use 64 1×1 kernels. This way, the non-linear mapping is performed on the pixel itself, without considering the relationships among neighbors.
- 3) **Reconstruction layer.** The final layer is designed for reconstructing a colorful patch from a given set of features. The kernels are 5×5 and initialized as Gaussian kernels, exploiting the local information to reconstruct the final colors.

ReLU (Rectified Linear Units, $\max(0, x)$) [39] is used as the activation function as it can often avoid the gradient vanishing/exploding problem. Mathematically, the network can be formulated as :

$$F_1(\mathbf{Y}) = \max(0, W_1 * \mathbf{Y} + B_1), \quad (1)$$

$$F_2(\mathbf{Y}) = \max(0, W_2 * F_1(\mathbf{Y}) + B_2), \quad (2)$$

$$F(\mathbf{Y}) = W_3 * F_2(\mathbf{Y}) + B_3, \quad (3)$$

where \mathbf{Y} is the input patch; F_i is the output feature map of the i th layer; W_i and B_i respectively represent the filters and the bias vector of the i th layer; and $*$ is the convolution operator.

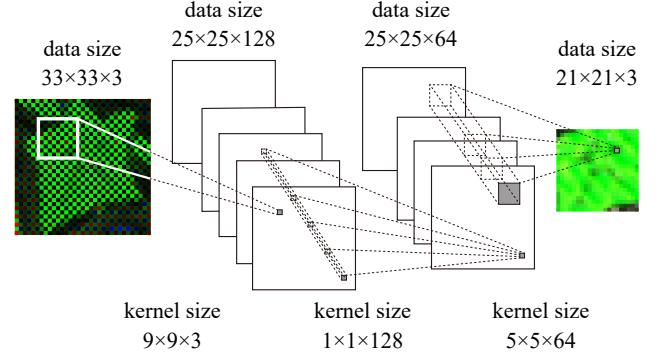


Fig. 1: The architecture of DMCNN. The input 33×33 mosaiced image patch sampled with the Bayer CFA is first extended to $33 \times 33 \times 3$ by adding zeros for missing channels. It then goes through three stages, feature extraction, non-linear mapping and reconstruction. Finally, the network outputs a reconstructed patch with three color channels.

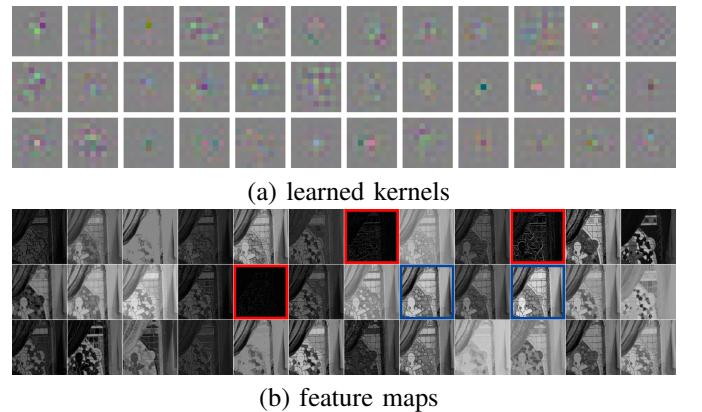


Fig. 2: Visualization of learned kernels and feature maps for an example. (a) 36 of 128 kernels in the first convolutional layer. (b) Corresponding feature maps.

Let $\Theta = \{W_1, W_2, W_3, B_1, B_2, B_3\}$ denote parameters of the DMCNN network. The L_2 norm is used as the loss function.

$$L(\Theta) = \frac{1}{n} \sum_{i=1}^n \|F(\mathbf{Y}_i; \Theta) - \mathbf{X}_i\|^2, \quad (4)$$

where \mathbf{Y}_i is the i th mosaiced patch and \mathbf{X}_i is its corresponding colorful patch (ground truth); and n is the number of training samples. The stochastic gradient descent is used for finding the optimal parameters Θ . The learning rate is 1 for the first two layers while a smaller learning rate (0.1) is used for the last layer.

The DMCNN network is an end-to-end learning model with two advantages compared to previous demosaicing algorithms. First, the features are explored automatically and optimized in a data-driven manner rather than handcrafted. Second, the reconstruction could exploit more complicated spatial and spectral relationships. Fig. 2 visualizes some learned kernels and corresponding feature maps in the first convolutional layer for an image. It can be observed that some automatically

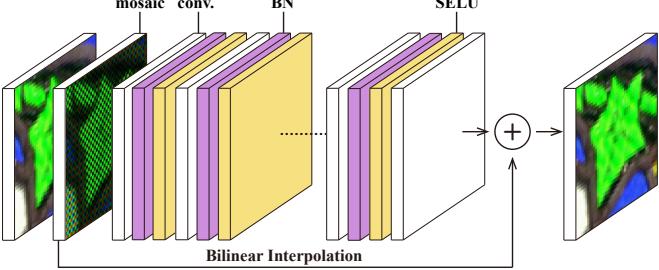


Fig. 3: The architecture of DMCNN-VD. It consists of 20 layers with the residual learning strategy. Each layer is composed of a convolution layer, a batch normalization layer and a SELU activation layer.

learned features explore directional information, which is often considered useful for demosaicing. For example, the 7th, 10th and 17th features outlined with red in Fig. 2(b) are gradient-like features with different orientations. It can also be found that some features are for chromatic interpolation, such as the 18th and the 20th features outlined with blue in Fig. 2(b). Such features could be difficult to design manually, but can be automatically constructed using CNN.

B. Very Deep DMCNN (DMCNN-VD)

Although DMCNN exploits the possibility of learning an end-to-end CNN model for demosaicing, it does not fully explore the potential of CNN models as the model is considerably shallow. It has been shown in many applications that, given the same number of neurons, a deeper neural network is often more powerful than a shallow one. Recently, residual learning has been shown effective on training deep networks with fast convergence [40]. Residual learning converges faster by learning the residual information and constructing the final solution by adding the learned residual information to the input. Kim et al. adopted the residual learning approach and proposed a much deeper end-to-end CNN architecture for super-resolution, VDSR [22]. Inspired by their model, we propose a design of a deep CNN model for demosaicing, very deep DMCNN (DMCNN-VD).

Fig. 3 illustrates the architecture of DMCNN-VD. It consists of N layers ($N = 20$ in our current setting). Each layer is composed of a convolution layer, a batch normalization layer [41] and a SELU activation layer [42]. Mathematically, the DMCNN-VD network is formulated as :

$$F_n(\mathbf{Y}) = \text{selu}(W_n * \mathbf{Y} + B_n), n = 1 \dots N-1 \quad (5)$$

$$F(\mathbf{Y}) = W_N * F_{N-1}(\mathbf{Y}) + B_N, \quad (6)$$

where $\text{selu}(x) = \lambda x$ if $x > 0$ and $\lambda\alpha(e^x - 1)$ otherwise. λ and α are constants defined in the SELU paper [42]. The loss function is also evaluated using the L_2 -loss form but with some differences from Equation (4):

$$L(\Theta) = \frac{1}{n} \sum_{i=1}^n \| (F(\mathbf{Y}_i; \Theta) + \tilde{\mathbf{Y}}_i) - \mathbf{X}_i \|^2. \quad (7)$$

	DMCNN	DMCNN-VD
Reference architecture	SRCNN [21]	VDSR [22]
ConvLayers	3	20
Activation function	ReLU	SELU
Kernel size	9-1-5	3
Feature map channel	128-64-3	64-64-...-3
Padding zero (pixel)	0	1
Gradient Updating	Clipping	Adam [43]
Residual Learning	No	Yes
Initialization	Gaussian	MSRA [40]

TABLE I: Details for the two proposed CNN architectures for demosaicing, DMCNN and DMCNN-VD.

The output of DMCNN-VD, $F(\mathbf{Y}_i; \Theta)$, refers to the residual between the ground truth patch \mathbf{X}_i and $\tilde{\mathbf{Y}}_i$, the bilinear interpolation of the input patch \mathbf{Y}_i . This way, the DMCNN-VD model only focuses on learning the differences between the ground truth and the baseline, often corresponding to the more difficult parts to handle. Thus, its learning can be more effective and efficient. In principle, any demosaicing method could be used to generate the input patch. Although bilinear interpolation could suffer from severe zipper and false color artifacts, it performs as well as the state-of-the-art methods on smooth areas which often represent a large portion of an image. In addition, as the method is simple, the artifacts tend to be more coherent and the residual information is easier to learn by CNN. Advanced method could produce sophisticated artifacts that are more difficult to learn. We found the results of bilinear interpolation are sufficient for residual learning. It also has the advantage of being more efficient than other alternatives.

Unless otherwise specified, we used 3×3 kernels and 1-pixel padding for all the intermediate layers. The MSRA initialization policy [40] was used for initialization. We used 0.001 as the factor of standard deviation. Adam [43] was adopted for gradient updating and we set the learning rate to $1e-5$. TABLE I gives details for the two proposed demosaicing architectures, DMCNN and DMCNN-VD.

C. Experiments with Bayer demosaicing

Benchmarks. The most popular benchmarks for demosaicing are the Kodak dataset and the McMaster dataset. All images in the Kodak dataset were captured by film cameras, scanned and then stored digitally. The dataset contains several challenging cases with high-frequency patterns that are difficult to be recovered from the samples of regular CFA patterns, such as the Bayer pattern. Zhang et al. [49] and Buades et al. [14] noticed that the images in the Kodak dataset tend to have strong spectral correlations, lower saturation, and smaller chromatic gradients than normal natural images. Thus, Zhang et al. introduced the McMaster benchmark (McM for short) which contains images with statistics closer to natural images [50]. Since both datasets present their own challenges, demosaicing algorithms are often evaluated on both of them. We follow the convention used in most of previous studies by using 12 Kodak images and 18 McM images as the evaluation benchmark.

Training set. The training data plays an important role in machine learning. However, we found the training data used

Algorithm	Kodak (12 photos)			McM (18 photos)			Kodak+McM (30 photos)					
	PSNR			CPSNR	PSNR			CPSNR	PSNR			
	R	G	B		R	G	B		R	G	B	
SA [44]	39.8	43.31	39.5	40.54	32.73	34.73	32.1	32.98	35.56	38.16	35.06	36.01
SSD [14]	38.83	40.51	39.08	39.4	35.02	38.27	33.8	35.23	36.54	39.16	35.91	36.9
NLS [16]	42.34	45.68	41.57	42.85	36.02	38.81	34.71	36.15	38.55	41.56	37.46	38.83
CS [45]	41.01	44.17	40.12	41.43	35.56	38.84	34.58	35.92	37.74	40.97	36.8	38.12
ECC [46]	39.87	42.17	39	40.14	36.67	39.99	35.31	36.78	37.95	40.86	36.79	38.12
RI [8]	39.64	42.17	38.87	39.99	36.07	39.99	35.35	36.48	37.5	40.86	36.76	37.88
MLRI [9]	40.53	42.91	39.82	40.88	36.32	39.87	35.35	36.60	38.00	41.08	37.13	38.32
ARI [10]	40.75	43.59	40.16	41.25	37.39	40.68	36.03	37.49	38.73	41.84	37.68	39.00
PAMD [47]	41.88	45.21	41.23	42.44	34.12	36.88	33.31	34.48	37.22	40.21	36.48	37.66
AICC [48]	42.04	44.51	40.57	42.07	35.66	39.21	34.34	35.86	38.21	41.33	36.83	38.34
DMCNN	39.86	42.97	39.18	40.37	36.50	39.34	35.21	36.62	37.85	40.79	36.79	38.12
DMCNN-VD	43.28	46.10	41.99	43.45	39.69	42.53	37.76	39.45	41.13	43.96	39.45	41.05

TABLE II: Quantitative evaluation. We compared our CNN-based methods (DMCNN and DMCNN-VD) with SA [44], SSD [14], NLS [16], CS [45], ECC [46], RI [8], MLRI [9], ARI [10], PAMD [47], AICC [48]. The best method is highlighted in red and the second best is highlighted in green in each category (column).

in previous demosaicing methods could be problematic. For example, the PASCAL VOC'07 dataset was adopted in previous work [16] and it has the following problems: (1) the images are of low quality which makes some demosaicing artifacts unavoidable, not to mention the compression artifacts in them; (2) the dataset was collected for object classification and the limited categories of image contents put restrictions, such as the one on the color distribution of images.

For the purpose of training image demosaicing methods, we collected 500 images from Flickr with following criteria: (1) the images are colorful enough to explore the color distributions in real world as much as possible; (2) there are high-frequency patterns in the images so that CNN learns to extract useful features for challenging cases; and (3) they are of high quality so that the artifacts due to noise and compression can be avoided as much as possible. The collected images were resized to roughly 640×480 to have more high-frequency patterns and at the same time, more likely to be mosaic-free. We call the dataset Flickr500. The images were rotated by 90° , 180° , 270° and flipped in each of directions for data augmentation. We extracted roughly 3.5 million patches from these images and used them for training the CNN models unless specified otherwise. The Flickr500 dataset and source codes will be released so that others can reproduce our work¹.

Quantitative comparison. We quantitatively compare the two proposed CNN models with ten existing algorithms, including SA [44], SSD [14], CS [45], ECC [46], AICC [48], three residual-interpolation-based methods (RI [8], MLRI [9] and ARI [10]) and two sparse-coding-based methods (NLS [16] and PAMD [47]). Following the convention, we use the PSNR (Peak signal-to-noise ratio) value as the metric. TABLE II summarizes the results of the quantitative comparison on the Kodak dataset, the McM dataset and their combination. Note that all numbers in TABLE II are directly adopted from previous work [8], [10] except DMCNN and DMCNN-VD. Thus, we followed the same setting with 12 Kodak images and 18 McM images when obtaining the numbers for DMCNN and DMCNN-VD. In each category (a column in the table), the best result is highlighted in red and the second best one in green. In most cases, we use the CPSNR value on

the combined dataset (Kodak+McM) as the final metric. The DMCNN model is competitive with the 38.12dB CPSNR value. However, it is outperformed by the best of ten previous methods, ARI [10], by almost 1 dB. The shallower layers without the residual-learning strategy makes it difficult to recover local details. On the other hand, with a deeper structure and the residual-learning model, DMCNN-VD reaches 41.05dB in CPSNR and outperforms all competing algorithms by a margin, 2.05dB better than the closest competitor ARI.

One thing to note is that, both NLS and our methods are learning-based. NLS was trained on the PASCAL VOC 2017 dataset while ours were trained on the Flickr500 dataset. To make a fair comparison, we trained DMCNN-VD on the PASCAL VOC 2007 dataset. The CPSNR values are 44.26, 37.35 and 40.11 for Kodak, McM and Kodak+McM respectively while NLS achieves 42.85, 36.15 and 38.83. The DMCNN-VD model still outperforms NLS by a margin. In addition, NLS requires expensive online learning and extra grouping for exploiting sparse coding and self-similarity. Thus, it is less efficient. On a PC with an Intel Core i7-4790 CPU and NVIDIA GTX 970 GPU, for demosaicing a 500×500 image, DMCNN took 0.12 second and DMCNN-VD took 0.4 second while NLS took roughly 400 seconds. Note that the CNN models ran with GPUs while NLS only used a CPU. It is not clear how much NLS could be accelerated with parallel computation.

Qualitative comparison. Fig. 4 shows visual comparisons on several examples. Some models in Fig. 4 will be discussed in Section IV. Fig. 4(a) gives an example from the McM dataset. Most previous methods and the DMCNN model cannot handle such saturated colors and thus produce extra diagonal stripes in the green star. On the contrary, the DMCNN-VD model performs much better thanks to its deeper architecture through the residual learning scheme. Fig. 4(b) shows another example from the McM dataset. The close-up shows a high-frequency regular pattern, which is difficult to recover for most previous algorithms. For example, ARI [10] generates noisy patterns in this case. The DMCNN-VD model gives a much better result. Fig. 4(c) gives an example from the Kodak dataset. The close-up shows the blind of the building, containing nearly horizontal stripes. In this case, residual-interpolated-based methods introduce significant false color artifacts, and

¹<http://www.cmlab.csie.ntu.edu.tw/project/Deep-Demosaic/>

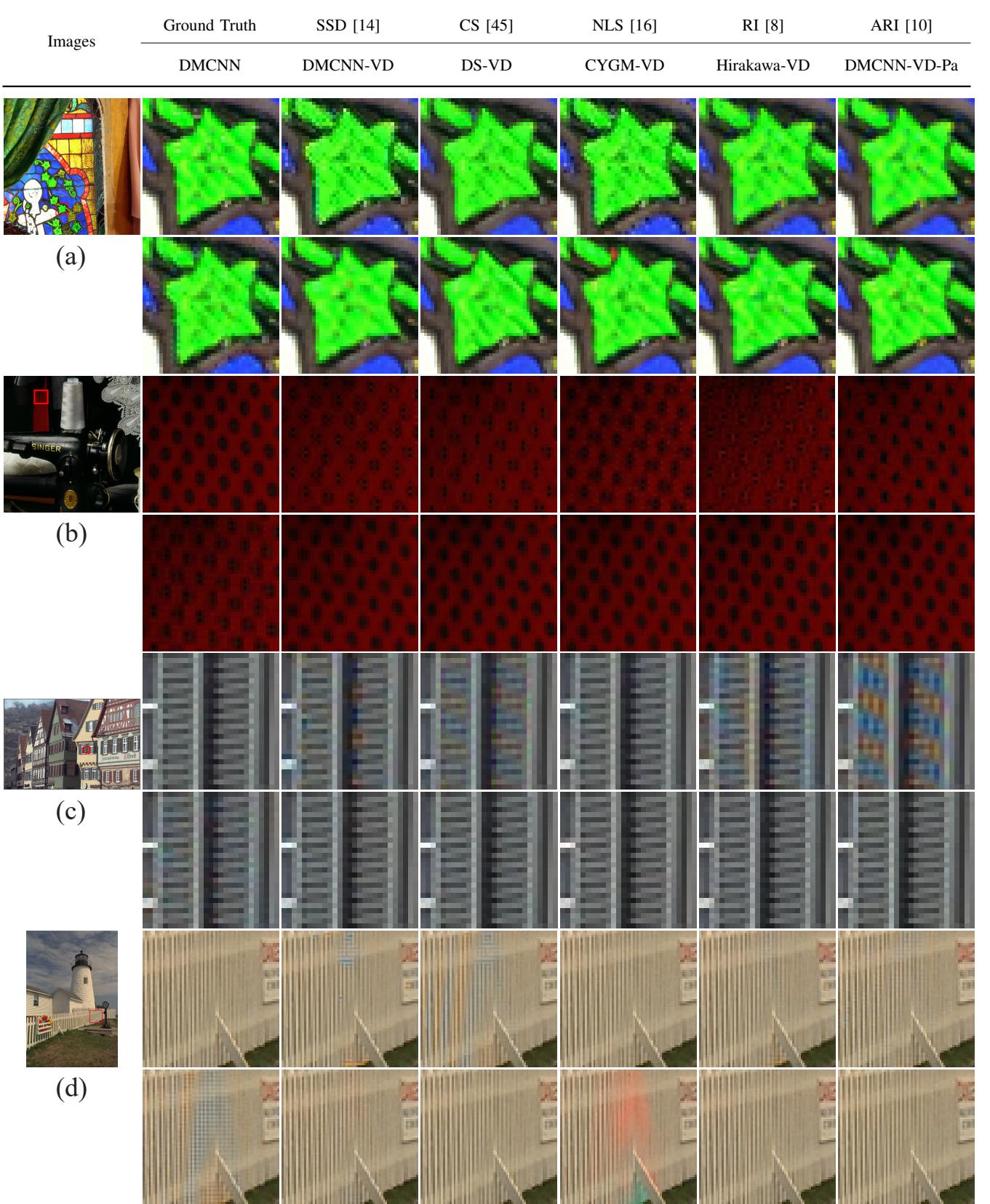


Fig. 4: Visual comparisons for color demosaicing. (a)(b) Examples from the McM dataset. (c)(d) Examples from the Kodak dataset. DS-VD, CYGM-VD and Hirakawa-VD are results of DMCNN-VD with diagonal stripe, CYGM and Hirakawa CFAs respectively. DMCNN-VD-Pa represents the DMCNN-VD model that learns CFA design and demosaicing jointly.

models	Kodak24	McM	average
SIGGRAPH Asia [37]	41.20	39.50	40.47
ICME [38]	42.04	38.98	40.73
DMCNN-VD (3×3)	41.85	39.45	40.82
DMCNN-VD (5×5)	42.19	39.56	41.06
DMCNN-VD (7×7)	42.36	39.74	41.24
DMCNN-VD on WED	42.47	39.54	41.21

TABLE III: Quantitative comparisons of different deep models on Bayer demosaicing, in terms of the average CPSNR values for the Kodak24 and the McM datasets. Note that, for direct comparison with other CNN models [37], [38], the setting for the Kodak dataset is different from the one used in TABLE II.

models	100 WED images	all WED images
ICME [38]	39.67	-
DMCNN-VD on Flickr500	40.94	40.18
DMCNN-VD on WED	41.55	-

TABLE IV: Quantitative comparisons with the ICME model on the WED dataset. The models were tested on 100 WED images. The DMCNN-VD model trained on the Flickr500 dataset is also tested on all 4,744 WED images.

so do SSD [14] and CS [45]. NLS [16] and the DMCNN-VD model have recovered the structure better, showing that such data-driven, automatically learned features can be more effective. In Fig. 4(d), we can observe that the high-frequency structure of the fence is very difficult for all methods to reconstruct perfectly. Artifacts like horizontal stripes can be found apparently in the results of interpolation-based methods. The only successful one is the NLS method [16] which could benefit from its self-similarity strategy.

Comparisons with other deep demosaicing methods. As mentioned in Section II, there were a couple of previous papers on deep Bayer demosaicing, one published in SIGGRAPH Asia 2016 [37] and the other in ICME 2017 [38]. It is difficult to compare with these methods fairly since the training sets are different and the source codes are not always available. The SIGGRAPH Asia 2016 model was trained on 2,590,186 128×128 difficult patches. The ICME 2017 model was trained with 384,000 50×50 patches extracted from 4,644 images from the Waterloo Exploration Dataset (WED) [51]. The first two rows of TABLE III shows the performance of previous work on the Kodak24 and the McM testing datasets, adopted directly from their papers. We tested the DMCNN-VD model with the same testing setting as theirs. The third row of TABLE III reports our results. With the default kernel size (3×3), the DMCNN-VD model has a slight advantage on the average CPSNR value.

The kernel size has impacts on the performance of the model. We experimented with different kernel sizes, 3×3 , 5×5 and 7×7 , for the DMCNN-VD model. TABLE III reports the results. It is clear that the performance improves with the kernel size. With the 7×7 kernel, the proposed model achieves the best performance at 42.36dB and 39.74dB for Kodak24 and McM respectively. However, a larger kernel size will also incur more computation cost on both training and testing. In the paper, without otherwise specified, we report the results with the 3×3 kernel.

Algorithms	Linear	sRGB
ARI [10]	39.94	33.26
RTF [52]	39.39	32.63
DMCNN-VD	41.35	34.78

TABLE V: Quantitative evaluation on the clean data of MDD. We compare our DMCNN-VD model to ARI [10] and RTF [52] by reporting CPSNR values.

To verify the proposed model with larger datasets, we applied the DMCNN-VD model trained on the Flickr500 dataset to the WED dataset. The WED dataset contains 4,744 images. The DMCNN-VD model achieves 40.18dB in terms of CPSNR. It shows the DMCNN-VD model can generalize very well. In addition, we have trained the DMCNN-VD model using the WED dataset. We used the same setting as the ICME paper in which 4,644 images were used for training and the rest 100 images for testing. The last row of TABLE III reports the results. When trained on the same dataset, the DMCNN-VD model achieves 42.27dB and 39.54dB for Kodak24 and McM respectively, outperforming the ICME model's 42.04dB and 38.98dB. When testing on 100 WED images, the DMCNN-VD (WED) model obtains 41.55dB while the ICME 2017 paper reports 39.67dB.

D. Experiments with the linear space and noise

Like most previous demosaicing papers, the previous section evaluates methods in the sRGB space. However, in real camera processing pipeline, the demosaicing process is often performed in the linear space of radiance rather than the sRGB space used in most demosaicing researches. This issue was recently addressed by Khashabi et al. [52]. They collected a new dataset called MDD (Microsoft Demosaicing Dataset). In this dataset, all images were captured by Canon 650D and Panasonic Lumix DMC-LX3. To simulate mosaic-free images, they proposed a novel down-sampling technique and converted data into the linear space. In addition, they also pointed out the input mosaiced images are usually noisy in reality. As the result, the dataset also provides noisy mosaiced images by adding noise extracted from the original raw images. In addition, they proposed a method for joint demosaicing and denoising by learning a nonparametric regression tree field (RTF) [52]. In the following experiments, we will first apply the pre-trained DMCNN-VD model directly to the MDD dataset and then improve its performance by transfer learning.

Clean data. The MDD dataset provides both clean and noisy mosaiced images. We first experiment with the clean versions for demosaicing. TABLE V reports the CPSNR values for three methods, ARI [10], RTF [52] and DMCNN-VD, in both the linear space and the sRGB space. Since RTF is trained with noisy inputs, it is not surprising that its performance on clean data is not as good as the state-of-the-art algorithm designed for clean inputs, ARI [10]. The proposed DMCNN-VD model performs very well with at least 1dB better than ARI in both spaces. Note that DMCNN-VD is trained in the sRGB space, but it still manages to perform well in the linear space.

Noisy data. The noise in the inputs could significantly hurt the performances of demosaicing methods, especially those

Algorithms	Panasonic (200)				Canon (57)			
	Linear		sRGB		Linear		sRGB	
	CPSNR	SSIM	CPSNR	SSIM	CPSNR	SSIM	CPSNR	SSIM
ARI [10]	37.29	0.956	30.59	0.871	40.06	0.97	31.82	0.886
RTF [52]	37.78	0.966	31.48	0.906	40.35	0.977	32.87	0.920
DMCNN-VD	38.33	0.968	32.00	0.920	40.99	0.978	32.89	0.927
DMCNN-VD-Tr	40.07	0.981	34.08	0.957	42.52	0.987	35.13	0.959

TABLE VI: Quantitative evaluation on the noisy data of MDD. We compare our DMCNN-VD and DMCNN-VD-Tr models to ARI [10] and RTF [52] by reporting both CPSNR and SSIM.

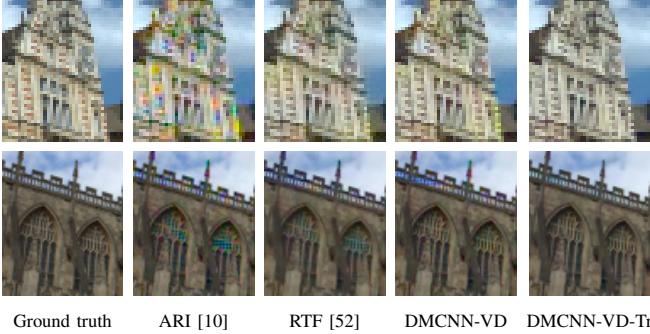


Fig. 5: Visual comparisons on MDD examples captured by Panasonic Lumix DMC-LX3 and Canon 650D.

deriving rules from clean inputs without taking noise into account. TABLE VI reports CPSNR and SSIM values in both the linear and sRGB spaces. Note that we report the results of Panasonic Lumix DMC-LX3 and Canon 650D separately in TABLE VI because they have difference noise characteristics. It confirms that the algorithm designed for clean data (ARI) could perform less well on noisy inputs. Although training on clean data, the proposed DMCNN-VD model performs surprisingly as well as the RTF method. Since noisy training data are available in MDD, we could leverage them for fine tuning the DMCNN-VD model trained on Flickr500 to improve its performance. This can be regarded as a transfer learning strategy [36]. In our case, the model transfers from the clean sRGB space to the noisy linear space. We denote the transferred model as DMCNN-VD-Tr. The CPSNR/SSIM values reported in TABLE VI show significant improvement by the fine-tuned DMCNN-VD-Tr model. Fig. 5 gives a couple of examples for visual comparisons. The top row shows an example with the Panasonic camera. Due to the noise presented in the input, the results of most algorithms are visually problematic, even the result of RTF shows perceivable color tinting. Such artifacts are hardly observable in the result of the fine-tuned DMCNN-VD-Tr model. The bottom row of Fig. 5 gives an example of the Canon camera. Again, the DMCNN-VD-Tr model recovers both color and structure information more faithfully than other methods. Fig. 6 shows visual comparisons of more examples on demosaicing in the noisy linear space. It is clear that the DMCNN-VD-Tr model performs joint demosaicing and denoising well, giving much better results than all other methods.

IV. DEMOSAICING WITH OTHER CFAS

In this section, we explore CNN models for demosaicing images with CFAs other than the Bayer one. We first apply

CNN to demosaicing with three other CFAs (Section IV-A). Next, we present a data-driven approach for joint optimization of the CFA design and the demosaicing method (Section IV-B). Finally, we apply the CNN model to a more challenging demosaicing problem with spatially varying exposure and color (Section IV-C).

A. Demosaicing with non-Bayer CFAs

Although the Bayer pattern is the most popular CFA, there are many other CFA designs. Fig. 7 shows three examples, *diagonal stripe* [53], *CYGM* and *Hirakawa* [18] CFAs. The diagonal stripe CFA (Fig. 7(a)) has a 3×3 unit pattern with the three primary colors uniformly distributed. The CYGM CFA (Fig. 7(b)) is proposed as it receives wider range of spectrum than the Bayer pattern. Its unit pattern is 2×2 with secondary colors and the green color. Several cameras have been built with this CFA. Finally, the Hirakawa CFA (Fig. 7(c)) was obtained by optimization through frequency analysis and has a 4×2 unit pattern.

Most demosaicing methods are bound up with specific CFAs. They would fail dramatically for other CFAs and often require complete redesigns to work with other CFAs. At the same time, most CFAs would require demosaicing methods specifically tailored for them for fully exploring their capability. One main strength of the demosaicing CNN model is its flexibility. The same CNN model can be used for different CFAs as long as it is re-trained with data encoded with the target CFAs. For a given CFA, the DMCNN-VD model is used while the input layer has to be adjusted with the CFA. As mentioned in Section III, the input layer consists of n color planes where n is the number of colors used in the CFA. For the Bayer CFA, three color planes are used because it consists of three primary colors. Taking the Hirakawa CFA as an example, its 4×2 tile consists of four colors, *deep pink*, *spring green*, *slate blue* and *chartreuse*. Thus, four color planes are used. For a pixel sampled with the deep pink channel, the sampled value is filled at the corresponding location of the deep pink color plane while the other three color planes are filled with zeros at the location. Three color planes are used for the diagonal stripe CFA and four for CYGM respectively.

TABLE VII reports performances of different combinations of CFAs and demosaicing algorithms. The first two rows show the performances of two state-of-the-art methods with the Bayer CFA, NLS and ARI, as the reference. The next four rows show the performances of the DMCNN-VD model with the Bayer CFA and the three CFAs in Fig. 7(a)-(c). For each CFA, the DMCNN-VD model is re-trained using the mosaic images sampled with the CFA. It is worth noting that, with the learned DMCNN-VD models, the Hirakawa CFA performs the best with 41.12dB, slightly better than the Bayer pattern with DMCNN-VD. It shows that a better pattern can improve demosaicing performance and the Hirakawa pattern could be the best CFA among the four CFAs experimented. However, although the Hirakawa pattern seems a better design, it is not easy to release its potential. For example, as shown in the second row from the bottom in TABLE VII, the Hirakawa CFA can only reach a mediocre performance at 37.23dB when

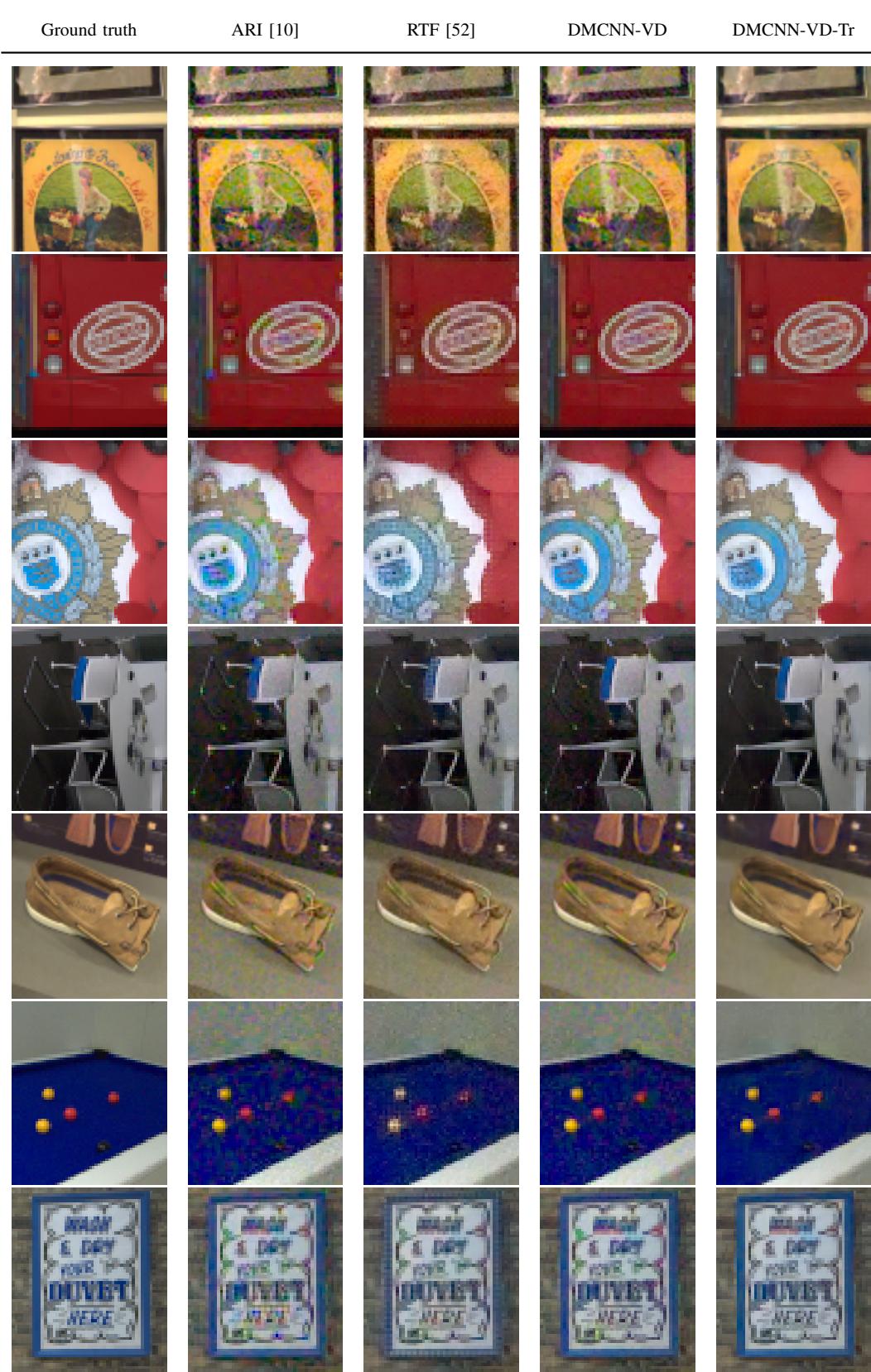


Fig. 6: Visual comparisons of demosaicing with noisy mosaiced images in the linear space. Both ARI and DMCNN-VD cannot handle noise well since they are trained on clear data. RTF performs better by taking advantages of noisy training data. By transfer learning, the DMCNN-VD-Tr model can perform joint denoising and demosaicing very well. It generates less noisy outputs with much less demosaicing artifacts.

Algorithm	Pattern	Kodak (12 photos)						McM (18 photos)						Kodak+McM (30 photos)					
		PSNR			CPSNR	PSNR			CPSNR	PSNR			CPSNR	PSNR			CPSNR		
		R	G	B		R	G	B		R	G	B		R	G	B			
NLS [16]	Bayer	42.34	45.68	41.57	42.85	36.02	38.81	34.71	36.15	38.55	41.56	37.46	38.83						
ARI [10]	Bayer	40.75	43.59	40.16	41.25	37.39	40.68	36.03	37.49	38.73	41.84	37.68	39.00						
DMCNN-VD	Bayer	43.28	46.10	41.99	43.45	39.69	42.53	37.76	39.45	41.13	43.96	39.45	41.05						
DMCNN-VD	Diagonal stripe	43.16	43.69	42.48	43.06	40.53	40.25	38.70	39.61	41.58	41.63	40.21	40.99						
DMCNN-VD	CYGM	40.78	45.83	42.40	42.50	39.16	42.33	38.70	39.73	39.81	43.73	40.18	40.84						
DMCNN-VD	Hirakawa	43.00	44.64	42.43	43.25	40.04	40.76	38.70	39.69	41.23	42.31	40.19	41.12						
Condat [23]	Hirakawa	41.99	43.18	41.53	42.16	33.93	34.83	33.44	33.94	37.15	38.17	36.68	37.23						
DMCNN-VD-Pa	Fig. 7(d)	43.42	43.80	42.59	43.23	40.96	40.44	39.02	39.98	41.95	41.79	40.45	41.28						

TABLE VII: Quantitative comparisons of demosaicing with different CFAs including Bayer CFA, diagonal stripe [53], CYGM, Hirakawa [18], and the proposed data-driven CFA.

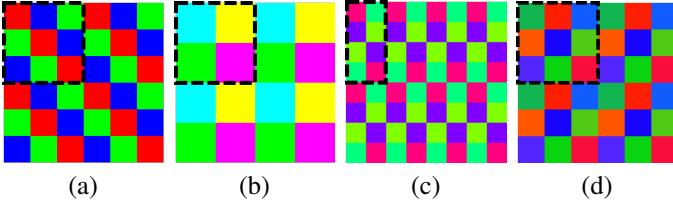


Fig. 7: Examples of different CFA designs: (a) Diagonal stripe [53], (b) CYGM and (c) Hirakawa [18] and (d) our CFA design found by the DMCNN-VD-Pa model.

demosiaicing with a previous method, Condat's algorithm [23]. It reveals that a good CFA design requires a good dedicated demosaicing method to work well. Since fewer methods were devised for CFAs other than the Bayer CFA, their potentials were not fully explored. The experiment shows how effective and flexible the CNN model is for demosaicing with different CFA designs. Fig. 4 shows visual comparisons on several examples. DS-VD, CYGM-VD and Hirakawa-VD are results of DMCNN-VD with diagonal stripe, CYGM and Hirakawa CFAs respectively.

B. Data-driven CFA design

From the previous section, we learn that the CFA design and the demosaicing algorithm have strong relationship and influence with each other. However, to the best of our knowledge, most demosaicing researches focus on either designing mosaic CFAs or devising demosaicing methods and there is no previous work that optimizes both jointly and simultaneously. Since the CNN model is effective and flexible on learning demosaicing algorithms for various CFAs, it is possible to embed the pattern design into the CNN model to simultaneously learn the CFA design and its demosaicing algorithm by joint optimization. The architecture is similar to an autoencoder which finds an effective and compact representation (encoding) for reconstructing the original image faithfully. In our case, the representation is formed by spatial color sampling/blending.

The pattern layer. We first introduce the pattern layer for forming a mosaic pattern. It is different from the popular CNN layers, such as convolution and pooling, available in deep learning frameworks. It cannot be composed using existing layers either. Thus, it has to be implemented as a new layer. Assume that the unit pattern is $m \times n$. That is, the unit pattern has $m \times n$ cells and each cell contains a color filter to convert

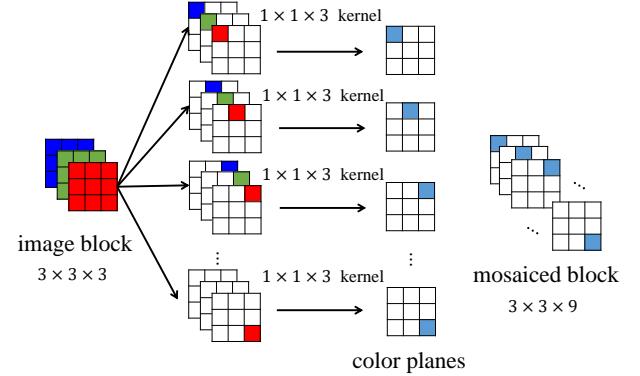


Fig. 8: The pattern layer. Assuming a 3×3 unit tile, we need nine color planes, each for a specific location. The layer converts an input $3 \times 3 \times 3$ patch into a $3 \times 3 \times 9$ patch which will be used as the input to the following DMCNN-VD model.

an RGB color into a value of some color channel. We can take the color filter as a $1 \times 1 \times 3$ filter kernel in the CNN model. Thus, we have to learn $mn 1 \times 1 \times 3$ kernels for a CFA design. Taking the Bayer CFA as an example, its unit pattern is 2×2 with four kernels $(1, 0, 0), (0, 1, 0), (0, 0, 1), (0, 1, 0)$ for $\mathbf{R}, \mathbf{G}_1, \mathbf{B}, \mathbf{G}_2$.

Fig. 8 shows an example of the pattern layer with a 3×3 unit pattern in the forward propagation pass. The input is a $3 \times 3 \times 3$ patch. For each cell, we have 1×3 filter to convert its RGB color into a value and put it on the corresponding cell. Since we have nine filters, the output consists nine color planes, each corresponding to a specific filter. Thus, the output of the pattern layer is $3 \times 3 \times 9$. Similar to the input structure used in DMCNN and DMCNN-VD models, each output pixel has nine channels. Among them, one is sampled and the other eight are left blank.

The DMCNN-VD-Pa model. We denote the CNN model with joint optimization of the mosaic pattern and the demosaicing algorithm by DMCNN-VD-Pa. Fig. 9 illustrates the DMCNN-VD-Pa model consisting of two major components.

- 1) *Pattern learning.* A pattern layer is responsible for learning color filters in CFA. In the forward pass, the pattern layer sub-sampled the full-color patch (ground truth) using its current filter kernels and outputs a multi-channel mosaiced patch. In the backward pass, gradients of the kernels are computed as normal convolution layers.

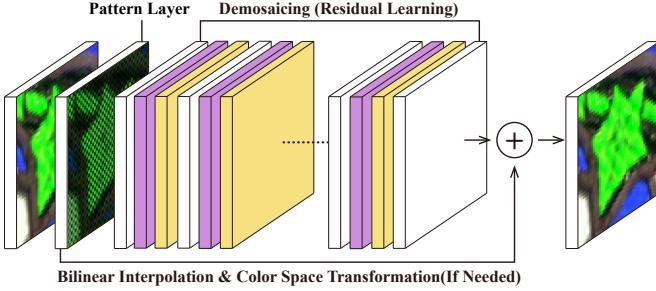


Fig. 9: The architecture of the DMCNN-VD-Pa model consisting of two stages: the pattern layer and the DMCNN-VD model. The baseline for residual learning is formed by bilinear interpolation and color transformation.

- 2) *Demosaicing*. The output of the pattern layer, nine color planes for a image patch with 8/9 of information missing, is used as the input to the demosaicing network. The DMCNN-VD model is used here for demosaicing. Note that, since the demosaicing network predicts missing information in all color planes, the output consists nine color planes with full information.

Assume that the nine kernels of the 3×3 unit tile are C_1, C_2, \dots, C_9 , each representing a RGB color. For residual learning, we first use bilinear interpolation to fill up each color plane. Thus, each pixel now has nine coefficients $\alpha_1, \dots, \alpha_9$, each for a color plane. We then transform the nine coefficients to a RGB color c by solving a linear system $Ac = b$ where A is a 9×3 matrix formed by stacking C_i row by row and b is the column vector $(\alpha_1, \dots, \alpha_9)^T$. The resultant RGB image is then used as the baseline for residual learning.

For training the above autoencoder-like CNN model, a set of images are taken as both inputs and labels. One thing to note is that the optimized pattern could include negative weights in the convolution kernels. Although optimal mathematically, kernels with negative weights are less practical for manufacturing. In addition, we would also like to limit the weights so that they are less than 1. Unfortunately, constrained optimization for CNN models is difficult. Similar to Chorowski and Zurada [54], we adopt the projected gradient descent algorithm [55] which projects gradients onto the feasible space for each update. In our case, for an optimal weight \tilde{w}_i found by regular gradient decent, we update the weights w_i of the CNN model as

$$w_i = \begin{cases} 0 & \text{if } \tilde{w}_i < 0 \\ 1 & \text{if } \tilde{w}_i > 1 \\ \tilde{w}_i & \text{otherwise.} \end{cases} \quad (8)$$

The weights are initialized with random numbers within $[0, 1]$ so that they start with a feasible solution. Fig. 7(d) shows the learned 3×3 pattern with a couple of interesting properties: (1) the pattern contains primary-color-like lights and (2) the arrangement of cells is regular and similar to the diagonal stripe pattern. It is worth noting that these properties are related to the chosen size of the unit pattern, 3×3 . For different sizes of unit patterns, the best pattern could have

different characteristics. Exploration with different pattern sizes is left as the future work.

Quantitative comparison. The last row of TABLE VII shows the performance of DMCNN-VD-Pa on the demosaicing benchmark. Its CPSNR value is 41.28dB on the combined dataset, more than 2.0dB better than ARI [10] and 0.23dB higher than the DMCNN-VD model with the Bayer CFA. The DMCNN-VD model with the Hirakawa pattern is the runner-up with 41.12dB. Note that the unit pattern of the Hirakawa pattern is 4×2 while the DMCNN-VD-Pa's is of 3×3 . It is also possible to use the proposed method for finding a good pattern with different tile sizes. Another interesting thing to note is that DMCNN-VD-Pa performs worse than DMCNN-VD on the green channel. It is reasonable since the Bayer CFA has doubled the samples in the green channel. By contrast, DMCNN-VD-Pa tends to sample three channels equally since the L_2 loss function simply averages over color channels. Since human is more sensitive to the green channel, to improve the perceptual quality, it is possible to increase the samples of green colors by altering the loss function with more emphasis on the green channel. They are left as future work.

Qualitative comparison. Fig. 4 shows the visual results of DMCNN-VD-Pa for several examples. Compared with the results of DMCNN-VD with the Bayer CFA, the new CFA helps correcting quite a few artifacts. For example, in Fig. 4(b), the result of DMCNN-VD-Pa is crisper and sharper than the one of DMCNN-VD. In Fig. 4(d), compared with DMCNN-VD, the zipper effect is almost completely removed by the new pattern of DMCNN-VD-Pa.

C. Demosaicing with spatially varying exposure and color

In addition to color demosaicing, the CNN model can also be applied to more general demosaicing problems. Here, we address the problem of demosaicing with spatially varying exposure and color (SVEC) sampling. More specifically, the CFA takes samples with different combinations of both colors and exposures. Fig. 10(a) gives a CFA design with three color channels, R, G and B, and two exposures, the low exposure e_1 and the high exposure e_2 (the high exposure is 64 times higher than the low one in our setting). It extends the Bayer CFA with spatially varying exposures. Fig. 10(b) and Fig. 10(c) show the images of the same scene captured with these two exposures. By taking pictures with the SVEC CFA in Fig. 10(a), it is possible to reconstruct a high dynamic range (HDR) image using only a single shot. However, the SVEC demosaicing problem is more challenging than color demosaicing since there is more information loss in SVEC demosaicing (5/6 of information is lost) than color demosaicing (2/3). Thanks to the flexible, end-to-end CNN model, we can address the more challenging SVEC demosaicing problem with the same models and proper training data. In this case, we have six channels in the input and the output is an HDR image with three color channels. Note that, rather than reconstructing six channels corresponding to RGB colors with two exposures, we directly recover real-valued RGB colors as the output.

Training data. For the problem setting of SVEC demosaicing, we need HDR images for simulating the captured images with



Fig. 10: The SVEC configuration. (a) The SVEC pattern used in the HDR experiment. (b) The image captured with the low exposure. (c) The image captured with the high exposure.

Algorithm	MSE	CPSNR
AP [25]	54.76	42.55
DMCNN	28.23	47.25
DMCNN-VD	14.89	53.13

TABLE VIII: Quantitative evaluation for SVEC demosaicing, in terms of the average MSE and CPSNR values for 50 testing HDR images.

different exposures. Unfortunately, HDR images often have quite different ranges and it could be problematic for training CNN models. To deal with this problem, we normalize the radiance images as

$$\tilde{I}_i = \lfloor \min\left(\frac{r_{max} - r_{min}}{I^{max} - I^{min}} * (I_i - I^{min}), r_{max}\right) \rfloor, \quad (9)$$

where \tilde{I}_i is the normalized radiance for the pixel i ; r_{max} and r_{min} are the maximum and minimum radiance values of the output range (2^{12} and 2^{-6} respectively in the current setting); I^{max} and I^{min} denote the maximum and minimum values of the original radiance image I ; the min and floor function simulates clamping and quantization of the camera pipeline (in our setting, the simulated sensor has 12 bits per pixel). After normalization, the SVEC pattern is applied to simulate the input. We collected 180 HDR images online and divided them into three subsets (100 for training, 30 for validation and 50 for testing) for the following experiments.

Quantitative comparison. For SVEC demosaicing, we compare our models to Assorted Pixel (AP) proposed by Nayar and Narasimhan [25] using MSE (mean square error) and CPSNR as metrics. TABLE VIII reports the results. The DMCNN model significantly outperforms AP in both metrics. With its deeper architecture, DMCNN-VD further improves the MSE error and the CPSNR value. It shows that the CNN models are more powerful than the simple regression model used by AP [25]. In addition, AP cannot capture the spatial relationships as well as the CNN models.

Qualitative comparison. Fig. 11 shows the SVEC demosaicing results for two testing images. For each example, we show the ground truth radiance maps and the radiance maps recovered by AP, DMCNN and DMCNN-VD, all visualized with the heat map. The difference maps show that the results of the DMCNN model are closer to the ground truth as it has more blue colors in the difference maps. With its deeper structure, DMCNN-VD further reduces the errors. The close-ups shows that the DMCNN model generates less artifacts around edges than AP while DMCNN-VD outperforms DMCNN with even sharper edges.

V. CONCLUSIONS

In this paper, we present a thorough study on applying the convolutional neural network to various demosaicing problems. Two CNN models, DMCNN and DMCNN-VD, are presented. Experimental results on popular benchmarks show that the learned CNN model outperforms the state-of-the-art demosaicing methods with the Bayer CFA by a margin, in either the sRGB space or the linear space. Experiments also show that the CNN model can perform demosaicing and denoising jointly. We also demonstrate that the CNN model is flexible and can be used for demosaicing with any CFA. For example, the current demosaicing methods with the Hirakawa CFA fall far behind the ones with the Bayer CFA. However, our learned CNN model with the Hirakawa CFA outperforms the-state-of-the-art methods with the Bayer CFA. It shows that the Hirakawa CFA could be a better pattern if a proper demosaicing method is employed. It shows the flexibility and effectiveness of the CNN model. We have also proposed a pattern layer and embedded it into the demosaicing network for joint optimization of the CFA pattern and the demosaicing algorithm. Finally, we have addressed a more general demosaicing problem with spatially varying exposure and color sampling. With the CNN model, it is possible to obtain a high dynamic range image with a single shot. All experiments show that the CNN model is a versatile and effective tool for demosaicing problems.

REFERENCES

- [1] N.-S. Syu, “Learning a deep convolutional network for demosaicking,” June 2016, Master Thesis, National Taiwan University.
- [2] B. K. Gunturk, J. Glotzbach, Y. Altunbasak, R. W. Schafer, and R. M. Mersereau, “Demosaicing: color filter array interpolation,” *IEEE Signal Processing Magazine*, vol. 22, no. 1, pp. 44–54, Jan 2005.
- [3] X. Li, B. Gunturk, and L. Zhang, “Image demosaicing: a systematic survey,” in *Electronic Imaging 2008*, vol. 6822, no. 1, Jan 2008.
- [4] D. Menon and G. Calvagno, “Color image demosaicking: An overview,” *Signal Processing: Image Communication*, vol. 26, no. 8, pp. 518 – 533, 2011.
- [5] B. Bayer, “Color imaging array,” 1976, US Patent 3,971,065.
- [6] D. R. Cok, “Signal processing method and apparatus for producing interpolated chrominance values in a sampled color image signal,” Feb. 10 1987, US Patent 4,642,678.
- [7] J. E. Adams Jr, “Interactions between color plane interpolation and other image processing functions in electronic photography,” in *IS&T/SPIE’s Symposium on Electronic Imaging: Science & Technology*. International Society for Optics and Photonics, 1995, pp. 144–151.
- [8] D. Kiku, Y. Monno, M. Tanaka, and M. Okutomi, “Residual interpolation for color image demosaicking,” in *Proceedings of IEEE ICIP 2013*, 2013, pp. 2304–2308.
- [9] ——, “Minimized-laplacian residual interpolation for color image demosaicking,” in *IS&T/SPIE Electronic Imaging*. International Society for Optics and Photonics, 2014, pp. 90.230L–90.230L.
- [10] Y. Monno, D. Kiku, M. Tanaka, and M. Okutomi, “Adaptive residual interpolation for color image demosaicking,” in *Proceedings of IEEE ICIP 2015*, 2015, pp. 3861–3865.
- [11] C. A. Laroche and M. A. Prescott, “Apparatus and method for adaptively interpolating a full color image utilizing chrominance gradients,” Dec. 13 1994, US Patent 5,373,322.
- [12] J. E. Adams Jr, “Design of practical color filter array interpolation algorithms for digital cameras,” in *Electronic Imaging ’97*. International Society for Optics and Photonics, 1997, pp. 117–125.
- [13] R. Kakarala and Z. Baharov, “Adaptive demosaicing with the principal vector method,” *IEEE Transactions on Consumer Electronics*, vol. 48, no. 4, pp. 932–937, 2002.
- [14] A. Buades, B. Coll, J.-M. Morel, and C. Sbert, “Self-similarity driven color demosaicking,” *IEEE Transactions on Image Processing*, vol. 18, no. 6, pp. 1192–1202, 2009.

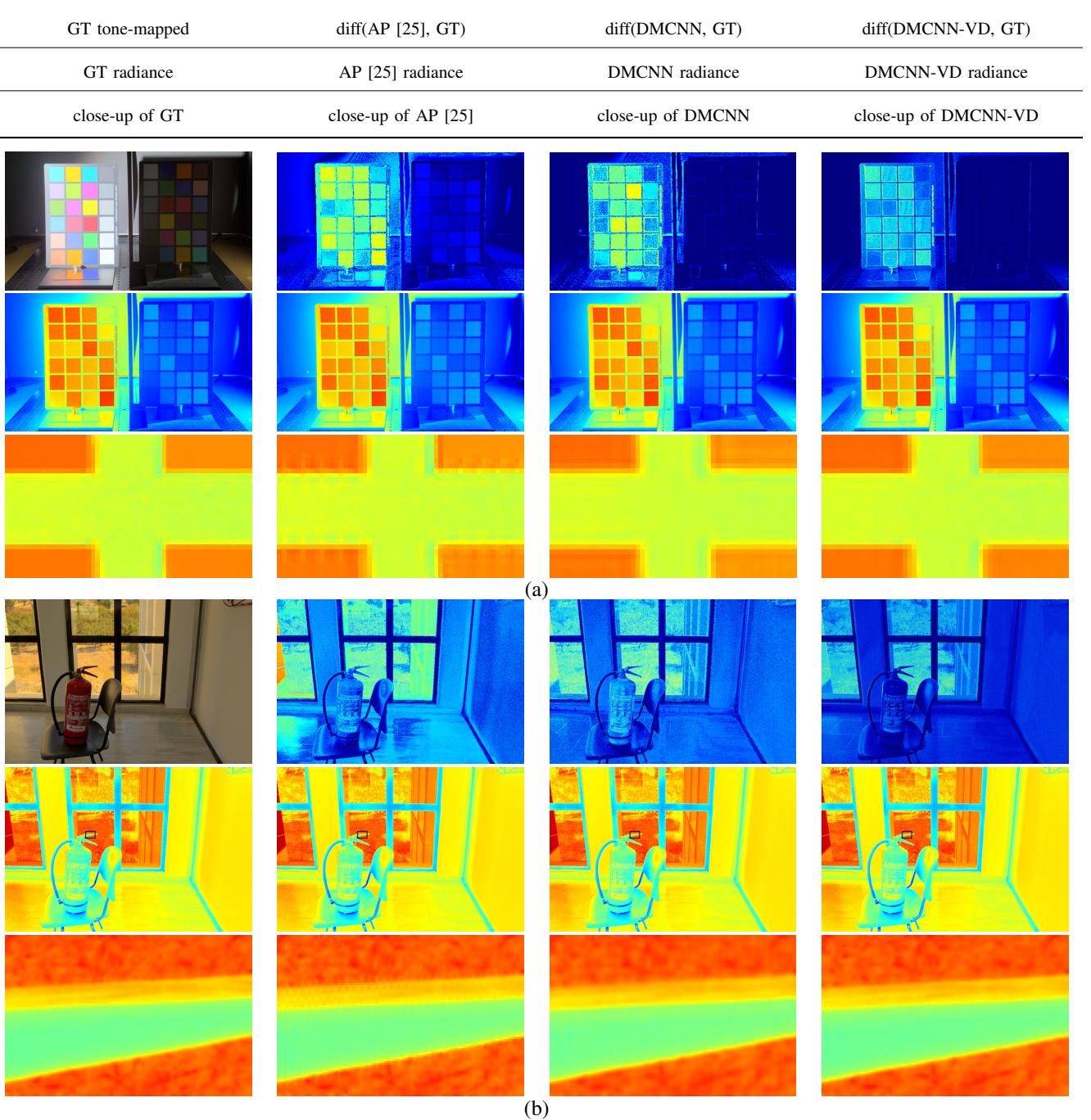


Fig. 11: Visual comparisons on SVEC demosaicing. GT means ground truth. DMCNN-VD and DMCNN have less errors as their difference maps contain more blue colors. In general, both perform better than AP with less artifacts around edges.

- [15] J. Mairal, M. Elad, and G. Sapiro, “Sparse representation for color image restoration,” *IEEE Transactions on Image Processing*, vol. 17, no. 1, pp. 53–69, 2008.
- [16] J. Mairal, F. Bach, J. Ponce, G. Sapiro, and A. Zisserman, “Non-local sparse models for image restoration,” in *Proceedings of IEEE ICCV 2009*, 2009, pp. 2272–2279.
- [17] D. Alleysson, S. Süsstrunk, and J. Hérault, “Linear demosaicing inspired by the human visual system,” *IEEE Transactions on Image Processing*, vol. 14, no. 4, pp. 439–449, 2005.
- [18] K. Hirakawa and P. J. Wolfe, “Spatio-spectral color filter array design for optimal image recovery,” *IEEE Transactions on Image Processing*, vol. 17, no. 10, pp. 1876–1890, 2008.
- [19] P. Hao, Y. Li, Z. Lin, and E. Dubois, “A geometric method for optimal design of color filter arrays,” *IEEE Transactions on Image Processing*, vol. 20, no. 3, pp. 709–722, 2011.
- [20] C. Bai, J. Li, Z. Lin, and J. Yu, “Automatic design of color filter arrays in the frequency domain,” *IEEE Transactions on Image Processing*, vol. 25, no. 4, pp. 1793–1807, 2016.
- [21] C. Dong, C. C. Loy, K. He, and X. Tang, “Learning a deep convolutional network for image super-resolution,” in *Proceedings of ECCV 2014*, 2014, pp. 184–199.
- [22] J. Kim, J. K. Lee, and K. M. Lee, “Accurate image super-resolution using very deep convolutional networks,” in *Proceedings of IEEE CVPR 2016*, 2016.
- [23] L. Condat, “A new color filter array with optimal properties for noiseless and noisy color image acquisition,” *IEEE Transactions on Image*

- Processing*, vol. 20, no. 8, pp. 2200–2210, 2011.
- [24] Y. Li, P. Hao, and Z. Lin, “The frequency structure matrix: A representation of color filter arrays,” *International Journal of Imaging Systems and Technology*, vol. 21, no. 1, pp. 101–106, 2011.
- [25] S. K. Nayar and S. G. Narasimhan, “Assorted pixels: Multi-sampled imaging with structural models,” in *Proceedings of ECCV 2002*, 2002, pp. 636–652.
- [26] F. Yasuma, T. Mitsunaga, D. Iso, and S. K. Nayar, “Generalized assorted pixel camera: postcapture control of resolution, dynamic range, and spectrum,” *IEEE Transactions on Image Processing*, vol. 19, no. 9, pp. 2241–2253, 2010.
- [27] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “Imagenet classification with deep convolutional neural networks,” in *Advances in neural information processing systems*, 2012, pp. 1097–1105.
- [28] C. J. Schuler, M. Hirsch, S. Harmeling, and B. Schölkopf, “Learning to deblur,” *CoRR*, vol. abs/1406.7444, 2014.
- [29] L. Xu, J. S. Ren, C. Liu, and J. Jia, “Deep convolutional neural network for image deconvolution,” in *Advances in Neural Information Processing Systems*, 2014, pp. 1790–1798.
- [30] J. Xie, L. Xu, and E. Chen, “Image denoising and inpainting with deep neural networks,” in *Advances in Neural Information Processing Systems*, 2012, pp. 341–349.
- [31] X. Zeng, W. Bian, W. Liu, J. Shen, and D. Tao, “Dictionary pair learning on grassmann manifolds for image denoising,” *IEEE Transactions on Image Processing*, vol. 24, no. 11, pp. 4556–4569, 2015.
- [32] J. Kim, J. K. Lee, and K. M. Lee, “Deeply-recursive convolutional network for image super-resolution,” in *Proceedings of IEEE CVPR 2016*, 2016.
- [33] R. Liao, X. Tao, R. Li, Z. Ma, and J. Jia, “Video super-resolution via deep draft-ensemble learning,” in *Proceedings of ICCV 2015*, 2015, pp. 531–539.
- [34] Z. Cheng, Q. Yang, and B. Sheng, “Deep colorization,” in *Proceedings of ICCV 2015*, 2015, pp. 415–423.
- [35] Z. Yan, H. Zhang, B. Wang, S. Paris, and Y. Yu, “Automatic photo adjustment using deep neural networks,” *ACM Transactions on Graphics*, vol. 35, no. 2, 2016.
- [36] C. Dong, Y. Deng, C. Change Loy, and X. Tang, “Compression artifacts reduction by a deep convolutional network,” in *Proceedings of ICCV 2015*, 2015, pp. 576–584.
- [37] M. Gharbi, G. Chaurasia, S. Paris, and F. Durand, “Deep joint demosaicking and denoising,” *ACM Trans. Graph.*, vol. 35, no. 6, pp. 191:1–191:12, 2016.
- [38] R. Tan, K. Zhang, W. Zuo, and L. Zhang, “Color image demosaicking via deep residual learning,” in *Proceedings of IEEE ICME 2017*, 2017.
- [39] V. Nair and G. E. Hinton, “Rectified linear units improve restricted boltzmann machines,” in *Proceedings of ICML 2010*, 2010, pp. 807–814.
- [40] K. He, X. Zhang, S. Ren, and J. Sun, “Delving deep into rectifiers: Surpassing human-level performance on imagenet classification,” in *Proceedings of the IEEE ICCV 2015*, 2015, pp. 1026–1034.
- [41] S. Ioffe and C. Szegedy, “Batch normalization: Accelerating deep network training by reducing internal covariate shift,” in *International Conference on Machine Learning (ICML)*, 2015, pp. 448–456.
- [42] G. Klambauer, T. Unterthiner, A. Mayr, and S. Hochreiter, “Self-normalizing neural networks,” in *Advances in neural information processing systems (NIPS)*, 2017.
- [43] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” *arXiv preprint arXiv:1412.6980*, 2014.
- [44] X. Li, “Demosaicing by successive approximation,” *IEEE Transactions on Image Processing*, vol. 14, no. 3, pp. 370–379, 2005.
- [45] P. Getreuer, “Image demosaicking with contour stencils,” *Image Processing on Line*, vol. 2, pp. 22–34, 2012.
- [46] S. P. Jaiswal, O. C. Au, V. Jakhetiya, Y. Yuan, and H. Yang, “Exploitation of inter-color correlation for color image demosaicking,” in *Proceedings of IEEE ICIP 2014*, 2014, pp. 1812–1816.
- [47] M. Zhang and L. Tao, “A patch aware multiple dictionary framework for demosaicing,” in *Proceedings of ACCV 2014*, 2014, pp. 236–251.
- [48] J. Duran and A. Buades, “A demosaicking algorithm with adaptive inter-channel correlation,” *Image Processing On Line*, vol. 5, pp. 311–327, 2015.
- [49] F. Zhang, X. Wu, X. Yang, W. Zhang, and L. Zhang, “Robust color demosaicing with adaptation to varying spectral correlations,” *IEEE Transactions on Image Processing*, vol. 18, no. 12, pp. 2706–2717, 2009.
- [50] L. Zhang, X. Wu, A. Buades, and X. Li, “Color demosaicing by local directional interpolation and nonlocal adaptive thresholding,” *Journal of Electronic Imaging*, vol. 20, no. 2, pp. 023016–023016, 2011.
- [51] K. Ma, Z. Duanmu, Q. Wu, Z. Wang, H. Yong, H. Li, and L. Zhang, “Waterloo exploration database: New challenges for image quality assessment models,” *IEEE Transactions on Image Processing*, vol. 26, no. 2, pp. 1004–1016, Feb 2017.
- [52] D. Khashabi, S. Nowozin, J. Jancsary, and A. W. Fitzgibbon, “Joint demosaicing and denoising via learned nonparametric random fields,” *IEEE Transactions on Image Processing*, vol. 23, no. 12, pp. 4968–4981, 2014.
- [53] R. Lukac and K. N. Plataniotis, “Color filter arrays: Design and performance analysis,” *IEEE Transactions on Consumer Electronics*, vol. 51, no. 4, pp. 1260–1267, 2005.
- [54] J. Chorowski and J. M. Zurada, “Learning understandable neural networks with nonnegative weight constraints,” *IEEE Transactions on Neural Networks and Learning Systems*, vol. 26, no. 1, pp. 62–69, 2015.
- [55] P. H. Calamai and J. J. Moré, “Projected gradient methods for linearly constrained problems,” *Mathematical programming*, vol. 39, no. 1, pp. 93–116, 1987.