Color Filter Array Demosaicking Using Densely Connected Residual Network

Bumjun Park¹ and Jechang Jeong², Member, IEEE
¹Department of Electronics and Computer Engineering, Hanyang University, Seoul, Republic of Korea
²Department of Electronic Engineering, Hanyang University, Seoul, Republic of Korea

Corresponding author: Jechang Jeong (e-mail: jjeong@hanyang.ac.kr).

“This work was supported by the research fund of Signal Intelligence Research Center supervised by Defense Acquisition Program Administration and Agency for Defense Development of Korea.”

ABSTRACT Deep convolutional neural networks have been used extensively in recent image processing research, exhibiting drastically improved performance. In this study, we apply convolutional neural networks to color filter array demosaicking, which plays an essential role in single-sensor digital cameras. Contrary to conventional convolutional neural network-based demosaicking models, the proposed model does not require any initial interpolation step for mosaicked input images, which increases the computational complexity. Using a mosaicked image as input, the proposed model is trained in an end-to-end manner to generate demosaicked images outputs. Many deep neural networks experience vanishing-gradient problem, which makes models hard to be trained. To solve this problem, we apply residual learning and densely connected convolutional neural network. Moreover, we apply block-wise convolutional neural networks to consider local features. Finally, we apply a sub-pixel interpolation layer to generate demosaicked output images more efficiently and accurately. Experimental results show that our proposed model outperforms conventional solutions and state-of-the-art models.

INDEX TERMS Demosaicking, color filter array interpolation, deep learning, convolutional neural network

I. INTRODUCTION

Digital color image pixels consist of three color components: red, green, and blue. To obtain exact information of these components, digital cameras need three sensors, which make cameras expensive and bulky. For this reason, most digital cameras use a single-sensor architecture with a color filter array (CFA). After penetrating the CFA, the sensor of the camera only takes one color information per pixel, according to the arrangement of the color in the CFA. There are many CFA patterns, but the Bayer pattern [1] is the most widely used. Figure 1 shows an example of the Bayer pattern. The number of green pixels in the Bayer pattern is twice that of red or blue pixels. This is because the human eye perceives spatial details chiefly from luminance information, and the luminosity function is similar to the CIE 1931 green matching function [2]. As there is only information of one color per pixel in single-sensor digital cameras, information of the other two colors should be interpolated. This interpolation process, referred to as demosaicking, is CFA interpolation.

Many demosaicking algorithms have been proposed [3]–[5]. In the past, demosaicking was implemented with simple interpolation algorithms: nearest neighbor [6], bilinear [7], or bicubic interpolation [8]. However, these simple interpolations resulted in many false color artifacts: blurring, chromatic aliases, zippering, and purple fringing [9]. To overcome this, Zhang et al. proposed color demosaicking via directional linear minimum mean square-error estimation (DLMMSE) [10]. Dengwen et al. proposed colour
demosaicking with directional filtering and weighting (DDFW) [11]. Pekkuuksen et al. proposed edge strength filter based demosaicking (ESF) [12] and multi gradients-based demosaicking (MSG) [13]. Recently, many demosaicking algorithms using residual interpolation have been proposed. Monno et al. proposed adaptive residual interpolation for demosaicking (ARI) [14] and Kim et al. proposed four-direction residual interpolation for demosaicking (FDIR) [15].

Despite many proposals, there still existed false color artifacts in the result images of conventional algorithms. Recently, deep convolutional neural networks (CNNs) have been applied to many image processing algorithms, including classification [16], [17], super resolution [18]–[20], high dynamic range [21], deblurring [22], denoising [23]–[25], dehazing [26], and deraining [27], and exhibited significantly improved performance. There are also many proposed demosaicking methods based on CNN architecture [28]–[30]. Tan et al. proposed color image demosaicking via deep residual learning (DDRL) [31] and Tan et al. proposed image demosaicking via multiple CNN (DDemo) [32].

Inspired by these CNN-based image processing solutions, we propose a CFA demosaicking method using a densely connected residual network (DRDN). Many conventional demosaicking solutions based on CNN architecture require substantial pre-processing: initial interpolation, initial demosaicking, and training image clustering [32]. Initial interpolation and initial demosaicking are applied by almost every conventional CNN-based demosaicking solution to equalize the resolution of the input image with the output image. Although they allow CNN models to easily generate output images with the desired resolution, they drastically increase the computational complexity and the memory use of the CNN model, as it receives three times more input data with two times higher resolution. Furthermore, the pre-processing method itself considerably consumes the computational complexity if conventional demosaicking algorithms are applied for their initial demosaicking process. The clustering of the training images also increases the computational complexity, as it needs to train the CNN model for each cluster, even after the clustering process.

To overcome these issues, our proposed model is trained in an end-to-end manner, which does not require any initial interpolation or demosaicking process to generate demosaicked output images with the desired resolution. Instead of the initial interpolation or demosaicking process, the proposed model divides the mosaicized image into four color layers, which are a quarter of the mosaicized image for the input of the proposed network. The layers consist of one red layer, one blue layer, and two green layers. There are twice as many green layers as red or blue layers, because the Bayer pattern [1] has twice as many green pixels as red or blue pixels. This input data modification enables our proposed network to consume less computational complexity and memory than conventional demosaicking networks. Then, we train our proposed CNN model to generate demosaicked images with the same size as the original mosaicced image. To generate demosaicked images more accurately and efficiently, we apply a sub-pixel interpolation layer [33] that learns to generate demosaicked images with desired resolution while our proposed CNN model is trained. Moreover, our proposed DRDN applies the residual learning [17] on the densely connected CNN [16] to avoid the vanishing-gradient problem, and block-wise CNN to consider local features. Finally, we apply a self-ensemble method [19], [34]. It enables our proposed network to exhibit better performance by applying an ensemble method without additional training or clustering processes.

The rest of this paper is organized as follows. Section II briefly describes the conventional networks that inspired us, and summarizes our contributions. Section III describes our proposed CNN model in detail. Section IV shows how we trained our proposed model and present the experimental results. Finally, section V outlines the conclusion of this study.

## II. RELATED WORKS

### A. CONVENTIONAL DEMOSAICKING NETWORKS

A lot of effort has been made to apply the CNN architecture to demosaicking. Tan et al. proposed the DDRL, which estimates the intermediate green channel to use it as a guidance for the reconstruction of the red and blue channels. However, the DDRL interpolates the input CFA image by bilinear interpolation [7] to make the resolution of the input image same with the desired output image resolution. This causes their network to consume more memory and computational complexity than they need, as they force their network to handle larger resolution feature maps. Tan et al. proposed the DDemo, which uses initial demosaicking and clustering methods for their network. As mentioned above, the initial demosaicking method itself consumes a lot of computational complexity. Besides, it also makes their network more computationally complex, and consumes more memory. Moreover, the clustering of the training images makes their method even more complex, because they need to train a model for each image cluster while clustering itself also consumes computational complexity. To reduce the unnecessary computational complexity, we modify the input mosaicced image into four color layers, which are a quarter the size of the original image. Because there are twice as many green pixels than red or blue pixels, we generate two green layers, one red layer, and one blue layer. Owing to the reduction of the input images resolution due to the input image modification, our proposed network can use memory efficiently, and considerably reduce the computational complexity.

### B. RESIDUAL DENSE NETWORK FOR IMAGE RESTORATION

He et al. proposed deep residual learning for image recognition (ResNet) [17]. As CNN architectures became deeper, many
models experienced the vanishing-gradient problem. ResNet solves this problem by applying a skip connection that enables the network to learn residual functions with reference to the input layers. In the following year, Huang et al. proposed densely connected convolutional networks (DenseNet) [16]. They connected each layer to every other layer in a feed-forward fashion to solve the vanishing-gradient problem and induce the network to reuse the information of the previous layers. Recently, Zhang et al. proposed residual dense network for image restoration (RDN) [20], which combines the idea of ResNet and DenseNet. They applied global residual learning to the DenseNet and local residual learning to the dense block to solve the vanishing-gradient problem. Inspired by these models, we propose the DRDN, which is optimized for demosaicking.

**C. SUB-PIXEL INTERPOLATION**

Shi et al. proposed super resolution using efficient sub-pixel convolutional neural network (ESPCN) [33]. Their proposed sub-pixel interpolation layer learns an upsampling filters to upscale their final output. This enables their network to reduce the computational complexity, as the input of their network is an original low resolution image instead of an interpolated high resolution image, owing to the sub-pixel interpolation layer. Inspired by this model, we apply their sub-pixel interpolation layer to reduce the computational complexity and increase accuracy of the interpolation.

**D. SELF-ENSEMBLE METHOD**

The self-ensemble method was introduced in [19] and [34]. In the conventional methods, they use a multi-model structure to apply the ensemble method [32]. However, the multi-model structure costs a lot of time and computational complexity, as it needs to train additional networks. On the other hand, the self-ensemble method, which averages the outputs of the transformed input images, only needs one trained network. By applying the self-ensemble method, we can increase the performance of our proposed network without training additional networks.

**E. CONTRIBUTIONS**

In this paper, we propose an input data modification method that enables our proposed network to avoid using initial interpolation or demosaicking processes, and reduce the computational complexity and memory consumption. Next, we propose a demosaicking network that combines the idea of ResNet [17] and DenseNet [16] to solve the vanishing-gradient problem while successfully interpolating the missing pixels of the mosaicked image. Moreover, our network applies the idea of the sub-pixel interpolation layer [33] to demosaicking solution, which enables our proposed network to generate demosaicked images more efficiently and accurately. Finally, we apply the self-ensemble method, which shows significant performance enhancement without additional time or computational complexity consumptions.

**III. PROPOSED METHOD**

Conventional demosaicking solutions that apply CNN architectures consist of two steps: initial interpolation or demosaicking and a CNN network that refines the result of the initial demosaicking. However, the initial interpolation or demosaicking of the input image not only consumes considerable computational complexity itself, but also increases the computational complexity and memory use of the CNN network. To solve this critical problem, we propose an input data modification method that enables our proposed network to avoid applying initial interpolation or demosaicking process. The proposed input data modification process is discussed in subsection A. After the modification of the input data, our proposed CNN model generates demosaicked images. The details of our proposed network will be discussed in subsection B. Figure 2 shows a data flow of the proposed demosaicking solution, and Figure 3 shows a comparison of the process between the conventional demosaicking solutions and the proposed method.

**A. INPUT DATA MODIFICATION**

As shown in Figure 1, mosaicked images with the Bayer pattern have information of one color channel per pixel, and there are twice as many green pixels as red or blue pixels. By reordering these pixels, we generate four color layers that have a quarter of the size of the original mosaicked image. Accordingly, our input data modification process generates two green layers, one red layer, and one blue layer. Using this input data modification, we can reduce the computational complexity and memory consumption of our demosaicking solution in two ways. First, our proposed demosaicking solution can avoid applying initial interpolation or demosaicking process, which considerably consumes computational complexity. Because our input data modification only needs a simple reordering process, we can achieve a drastic computational complexity reduction compared with conventional demosaicking methods. Second, because our input data modification makes our input image into a quarter of the size of the original image, our CNN model can reduce its computational complexity and memory.
consumption. Because the resolution of the input image is reduced, the size of the feature maps is also reduced, which results in a reduction of the memory consumption for storing the information of the feature maps. Moreover, the number of convolution operations also reduces, which determines the computational complexity and training time of the network. Therefore, we can reduce the computational complexity and memory consumption of our proposed network.

### B. NETWORK STRUCTURE

Inspired by [16], [17], and [20], our proposed DRDN consists of four parts: initial convolution block, densely connected residual blocks (DRBs), final convolution layer, and sub-pixel interpolation layer. Figure 4 shows the structure of our proposed DRDN, where conv1 indicates 1×1 convolution layers and conv3 indicates 3×3 convolution layers. Note that there exist activation functions after every convolution layer. The initial convolution block consists of two convolution layers: a 1×1 convolution layer and a 3×3 convolution layer. The 1×1 convolution layer generates 12 feature maps to apply global residual learning before the sub-pixel interpolation layer. Then, the 3×3 convolution layer generates 64 feature maps for DRB. After the convolution operations of the initial convolution block, our proposed DRDN proceeds N DRBs. The structure of the DRB will be discussed in subsection C.

Next, our proposed network proceeds to the final convolution layer, which produces the output of the DRBs into 12 feature maps for the sub-pixel interpolation layer. Finally, the sub-pixel interpolation layer generates demosaicked images with the desired resolution.

### C. DENSELY CONNECTED RESIDUAL BLOCK

Inspired by [16], [17], and [20], our proposed DRB consists of two parts: convolution blocks and a transition layer. Figure 5 shows the structure of our proposed DRB. Note that there exist activation functions after every convolution layer. The convolution blocks consist of two convolution layers, which are the same as the initial convolution block of the proposed network structure. However, in the convolution block, the number of filters of the convolution layers is different. According to [16], the 3×3 and 1×1 convolution layers generate k and 4k feature maps, respectively, where k denotes the growth rate of the DRB. In our proposed DRB, there exist three convolution blocks, whose outputs are connected by dense connectivity. After extracting the feature maps by the convolution blocks, the transition layer generates 64 feature maps to apply the local residual learning in the DRB. Table I summarizes the size of the convolution layers of our proposed DRDN. Finally, the output of the previous DRB is used for the input of the next DRB.
similar to the FHD resolution (1920×1080). The DIV2K architecture have been proposed, many datasets have been.

**IV. EXPERIMENTS**

**A. TRAINING DETAILS**

As many image processing solutions that apply CNN architectures have been proposed, many datasets have been used for training networks. Recently, Agustsson et al. released the DIV2K training and validation datasets [35], which include high quality images. The DIV2K training dataset consists of 800 images where the resolution of each image is similar to the FHD resolution (1920×1080). The DIV2K validation dataset consists of 100 images, where the resolution of each image is similar to the training dataset. Given the high quality of the images in the DIV2K, many state-of-the-art image processing methods use this dataset and show improved performance. Thus, we train our proposed network with the DIV2K training and validation sets. When training our network, we use patches that is extracted from the training dataset where the width and height of the patches are set to 64 pixels. To augment the training patches, we randomly rotate and flip the input patches before entering the proposed network. We set the batch size of the training patches to 64, and train our proposed network for 300 epochs. We use the Adam optimizer [36] with an initial learning rate of $10^{-4}$, and divided it by 10 for every 100 epochs. For the activation function, we used the leaky rectified linear unit (leaky ReLU), where $\alpha$ is set to 0.1. We use the mean square error for the loss function. We set the number of DRB (N) to 15 and the growth rate (k) to 32. For the test sets, we use the Kodak [37] and McMaster [38] dataset, which are widely used as test sets for.

<table>
<thead>
<tr>
<th>TABLE I. Size of the convolution layers of the proposed network.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Layers</td>
</tr>
<tr>
<td>-----------------------</td>
</tr>
<tr>
<td>Initial Convolution</td>
</tr>
<tr>
<td>Conv1</td>
</tr>
<tr>
<td>Conv2</td>
</tr>
<tr>
<td>Conv3</td>
</tr>
<tr>
<td>Transition</td>
</tr>
<tr>
<td>Final Convolution</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>TABLE II. PSNR (dB) and SSIM results of the conventional demosaicking methods and the proposed method for the Kodak dataset.</th>
</tr>
</thead>
<tbody>
<tr>
<td>PSNR</td>
</tr>
<tr>
<td>---------------------------------------</td>
</tr>
<tr>
<td>DLMSE</td>
</tr>
<tr>
<td>DDFW</td>
</tr>
<tr>
<td>ESF</td>
</tr>
<tr>
<td>MSG</td>
</tr>
<tr>
<td>ARI</td>
</tr>
<tr>
<td>FDRI</td>
</tr>
<tr>
<td>DDRL</td>
</tr>
<tr>
<td>DDeMo</td>
</tr>
<tr>
<td>DRDN</td>
</tr>
<tr>
<td>DRDN+</td>
</tr>
<tr>
<td>DLMSE</td>
</tr>
<tr>
<td>DDFW</td>
</tr>
<tr>
<td>ESF</td>
</tr>
<tr>
<td>MSG</td>
</tr>
<tr>
<td>ARI</td>
</tr>
<tr>
<td>FDRI</td>
</tr>
<tr>
<td>DDRL</td>
</tr>
<tr>
<td>DDeMo</td>
</tr>
<tr>
<td>DRDN</td>
</tr>
<tr>
<td>DRDN+</td>
</tr>
<tr>
<td>DLMSE</td>
</tr>
<tr>
<td>DDFW</td>
</tr>
<tr>
<td>ESF</td>
</tr>
<tr>
<td>MSG</td>
</tr>
<tr>
<td>ARI</td>
</tr>
<tr>
<td>FDRI</td>
</tr>
<tr>
<td>DDRL</td>
</tr>
<tr>
<td>DDeMo</td>
</tr>
<tr>
<td>DRDN</td>
</tr>
<tr>
<td>DRDN+</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>TABLE III. Average PSNR (dB) and SSIM results of the conventional demosaicking methods and the proposed method for the Kodak dataset.</th>
</tr>
</thead>
<tbody>
<tr>
<td>PSNR</td>
</tr>
<tr>
<td>---------------------------------------</td>
</tr>
<tr>
<td>DLMSE</td>
</tr>
<tr>
<td>DDFW</td>
</tr>
<tr>
<td>ESF</td>
</tr>
<tr>
<td>MSG</td>
</tr>
<tr>
<td>ARI</td>
</tr>
<tr>
<td>FDRI</td>
</tr>
<tr>
<td>DDRL</td>
</tr>
<tr>
<td>DDeMo</td>
</tr>
<tr>
<td>DRDN</td>
</tr>
<tr>
<td>DRDN+</td>
</tr>
<tr>
<td>DLMSE</td>
</tr>
<tr>
<td>DDFW</td>
</tr>
<tr>
<td>ESF</td>
</tr>
<tr>
<td>MSG</td>
</tr>
<tr>
<td>ARI</td>
</tr>
<tr>
<td>FDRI</td>
</tr>
<tr>
<td>DDRL</td>
</tr>
<tr>
<td>DDeMo</td>
</tr>
<tr>
<td>DRDN</td>
</tr>
<tr>
<td>DRDN+</td>
</tr>
</tbody>
</table>

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2019.2939578, IEEE Access
demosaicking solutions. The Kodak dataset consists of 24 images where the resolution of each image is 768x512. The McMaster dataset consists of 18 images where each image is extracted from high resolution images with a size of 500x500.

### B. PERFORMANCE COMPARISON

We compare the performance of our proposed network with six non-CNN-based algorithms (DLMMSE [10], DDFW [11], ESF [12], MSG [13], ARI [14], and FDRI [15]) and two state-of-the-art CNN-based demosaicking models (DDRL [31] and DDemo [32]). For the objective comparison, we calculate the color peak signal to noise ratio (CPSNR) [39] and structural similarity (SSIM) [40] of the result images. When calculating the CPSNR, we remove 10 pixels around the borders of the resulting images and ground truth images to avoid boundary artifacts. Table II and III show the objective comparison results of the Kodak dataset, and Table IV and V show the objective comparison results of the McMaster dataset. Note that DRDN+ denotes the result of the proposed network which employed the self-ensemble method [19], [34]. In the tables, we highlighted the highest value with red, and the second highest value with blue. As expected, the CNN-based demosaicking models show better performances for most of the sequences than the non-CNN based algorithms when comparing CPSNR and SSIM.

For the Kodak dataset in Table II, the proposed methods with and without the self-ensemble method show the highest and second highest results for most of the sequences. However, it is interesting to note that the non-CNN-based conventional methods show good performance on the SSIM for some sequences. This implies that there is room for further development of demosaicking methods that employ deep neural networks. From the average performances comparison in Table III, it can be seen that the proposed method outperforms the conventional methods by up to 4.66 dB in terms of CPSNR results, and by up to 0.0156 in terms of SSIM.
The proposed method achieves even better performance when it applies the self-ensemble method. The proposed method shows improved CPSNR and SSIM results up to 4.89 dB and 0.0160, respectively. For the McMaster dataset in Table IV, the proposed method and the proposed method with the self-ensemble method achieve the highest and the second highest performance for the most of the sequences. The DDRL, which applies deep neural networks for demosaicking shows good performance on the CPSNR for some sequences; however, overall, the proposed method exhibits a superior performance than the conventional methods. The proposed method outperforms the conventional methods by up to 5.05 dB in terms of average CPSNR, and by up to 0.0464 in terms of average SSIM as shown in Table V.

When the self-ensemble method is employed, the proposed method exhibits improved average CPSNR and SSIM up to 5.19 dB and 0.0471, respectively. These results show that our proposed method outperforms the conventional methods significantly, and exhibits even more enhanced performance by applying the self-ensemble method.

Figure 6 and 7 present the result images of the Kodak and McMaster datasets, respectively, for the subjective comparison. When comparing the performances of the demosaicking methods, it is important to compare whether there exist any artifacts such as zippering or false color artifacts. In Figure 6, we compare the result images of the Kodak dataset. As shown in the figure, the conventional demosaicking methods show false color artifacts, both with and without the CNN architecture. However, the proposed method interpolates the pixel values accurately and does not present any artifacts. Figure 7 presents the resulting images of the McMaster dataset. As shown in the figure, the conventional demosaicking methods produce false color artifacts in the form of black dots. However, the proposed method does not exhibit any false color artifacts, which establishes its excellent performance.

C. ABLATION STUDY
To exhibit the effect of global residual learning (GR), local residual learning (LR), and dense connectivity (CONC), we performed an ablation study. Table VI shows the results of the

<table>
<thead>
<tr>
<th></th>
<th>GR</th>
<th>LR</th>
<th>CONC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Kodak</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>McMaster</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>CPSNR (dB)</td>
<td>15.31 / 0.3045</td>
<td>27.03 / 0.8223</td>
<td>42.12 / 0.9882</td>
</tr>
<tr>
<td>SSIM</td>
<td>0.3045</td>
<td>0.8223</td>
<td>0.9882</td>
</tr>
</tbody>
</table>
D. Difference to RDN

Although the network architecture of our proposed method is similar to that of RDN [20], there are three major differences: First, the architecture of the blocks is different. Unlike RDN, that uses six $3\times3$ convolution layers in its blocks, our DRDN uses three sets of $1\times1$ and $3\times3$ convolution layers; our experimental analysis indicates that using $1\times1$ convolution layer achieves better demosaicking performance. Second, RDN concatenates the output information of every block at the end of the network, thereby consuming a lot of memory, producing only a slight increase in the demosaicking performance. On the contrary, our DRDN does not use the output information of every block at the end of the network. Finally, we applied a different number of the blocks and growth rates, which are optimized for demosaicking and memory consumption.

V. CONCLUSION

In this paper, we proposed a densely connected residual demosaicking network which successfully applied dense connectivity and residual learning on the demosaicking solution. Conventional demosaicking methods applied initial interpolation or demosaicking process, which made their method computationally complex. To solve this problem, our proposed network was trained in an end-to-end manner without any pre-processes. Moreover, our proposed network generated four color layers with a quarter of the size of the original mosaicked image by input data modification. This enabled our network to reduce the computational complexity and memory consumptions. Next, we trained our proposed network, which applied residual learning on densely connected CNN to avoid the vanishing-gradient problem. Additionally, our proposed network applied the sub-pixel interpolation layer which learns to generate demosaicked
images with desired resolution while our proposed CNN model is trained. This enabled our network to generate the demosaicked images more efficiently and accurately. Finally, we applied the self-ensemble method, which enabled our proposed network to achieve even better performances without additional training or clustering process. The experimental results exhibited that our proposed network outperforms the conventional demosaicking methods for both objective and subjective comparisons.

REFERENCES