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/TIP.2018.2803341, IEEE Transactions on Image Processing

DeepDemosaicking: Adaptive Image Demosaicking via Multiple Deep Fully Convolutional Networks

Daniel Stanley Tan, Wei-Yang Chen, and Kai-Lung Hua

***Abstract*—Convolutional neural networks are currently the state-of-the-art solution for a wide range of image processing tasks. Their deep architecture extracts low and high-level features from images, thus, improving the model's performance. In this paper, we propose a method for image demosaicking based on deep convolutional neural networks. Demosaicking is the task of reproducing full color images from incomplete images formed from overlaid color filter arrays on image sensors found in digital cameras. Instead of producing the output image directly, the proposed method divides the demosaicking task into an initial demosaicking step and a refinement step. The initial step produces a rough demosaicked image containing unwanted color artifacts. The refinement step then reduces these color artifacts using deep residual estimation and multi-model fusion producing a higher quality image. Experimental results show that the proposed method outperforms several existing and state-of-the-art methods in terms of both subjective and objective evaluations.**

***Index Terms*—Image demosaicking, deep convolutional networks, multi-model fusion.**

I. Introduction

A Single monochromatic image sensor in digital cameras can only capture one color information in each pixel. To capture color images, most digital cameras use color filter arrays (CFA) to produce a mosaic image, arranging pixels in pattern of alternating colors. The most commonly used pattern is the Bayer pattern [1] shown in Fig. 1. The Bayer pattern, which is based on the physiology of the human eye, consists of 25% red, 50% green, and 25% blue color values. Consequently, color demosaicking is necessary to recover the missing color information from the mosaic images in order to reconstruct the full three channel color image. Interpolation-based methods such as bilinear interpolation, bicubic interpolation, and spline interpolation, were the first methods proposed to address the demosaicking problem. These methods can estimate the missing color values but they also produce unwanted color artifacts such as zippering, chromatic aliases, purple fringing, and blurring. Numerous methods were proposed [2]–[15] to deal with false color artifacts and obtain high-resolution images. Residual interpolation methods proposed by Kiku et al. [10]–[13] provide higher quality images than simple interpolation-based methods showing the potential of residual estimation for image demosaicking.

In the recent years, convolutional neural network (CNN) based solutions have been the state-of-the-art in numerous computer vision and image processing tasks including image recognition [16], [17], colorization [18], [19], image segmentation [20], [21], contour detection [22], [23], super resolution [24], denoising [25], and many more [26]–[28]. There have been many architectures proposed to wire these convolutional neural networks [16], [18]–[23], [29]. Fully convolutional networks (FCN) which consists of only convolutional layers have been widely used because of its ability to handle arbitrary-sized inputs. Dong et al. [30], [31], and Kim et al. [32] used FCN based methods for image super-resolution to construct high-resolution images from low-resolution images. This inspired us to utilize an architecture based on fully convolutional network layers with residual estimation for image demosaicking.

We propose a deep fully convolutional network model for image demosaicking on the Bayer pattern color filter array. Our method applies a fast initial demosaicking [2] on an input image and refine it using a convolutional neural network that approximates the residuals between the initial interpolation and ground truth image. We also integrated ensemble techniques to make the model more robust by training different networks on different subsets of the data which we cleverly fuse together using a weighted double interpolation fusion. We extensively compared our model to both standard and state-of-the-art techniques. Our model achieved higher quality demosaicking based on average peak signal to noise ratios (PSNR) as well as visual comparisons.

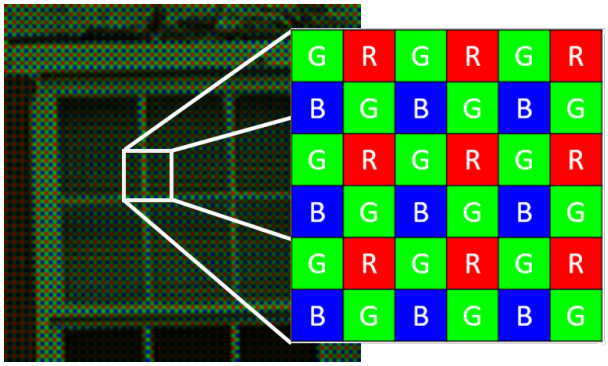


Fig. 1: This image shows the raw image data captured by an image sensor as well as the arrangement of the colors using the Bayer color filter array.

II. Related works

Many demosaicking methods have been proposed for the Bayer pattern color filter array. The early methods such as

D. S. Tan, W.-Y. Chen, and K.-L. Hua (corresponding) are with CSIE, National Taiwan University of Science and Technology, Taipei, Taiwan, ROC. (*Daniel Stanley Tan and Wei-Yang Chen contributed equally to this work.*) Corresponding e-mail: hua@mail.ntust.edu.tw.

1057-7149 (c) 2018 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

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/TIP.2018.2803341, IEEE Transactions on Image Processing

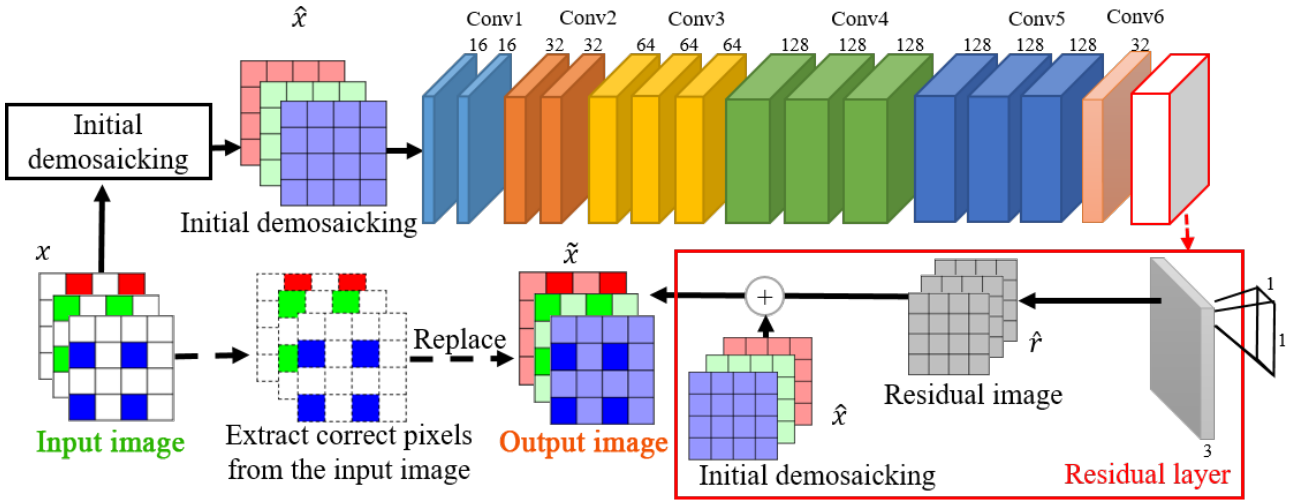


Fig. 2: Proposed network architecture of the DeepDemosaicking model. An initial demosaicked image $\hat{x}$ is used as an input to a series of fully convolutional layers (without the pooling layers) that represents the input image in a high dimnensional space. The residual layer then transforms it back into the original image dimensions and computes for the residuals from the ground truth image. Since the pixels from the original mosaic image already contains the correct values, we use them to replace the corresponding pixels in the output image. This architecture is inspired by the fully convolutional networks proposed by [20].

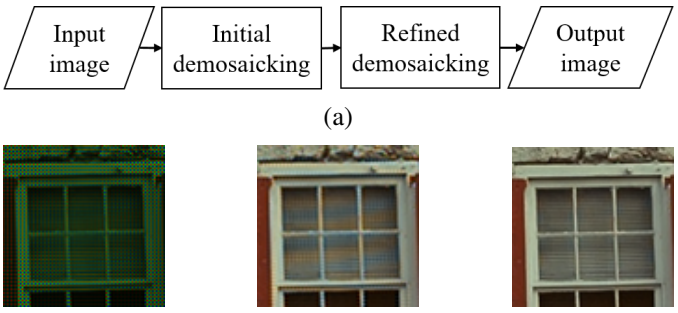


Fig. 3: This image shows an overview of the DeepDemosaicking process and its intermediate outputs. An initial demosaicking is first applied to the input image and then refined using CNN's to produce the final image.

bicubic interpolation, bilinear interpolation, and spline interpolation, treated each channel independently. The references [2], [33]–[40] used information found on other channels to assist in the reconstruction of the image. Gradient corrected methods leveraged on the image gradients to adjust the results [2], [33]. Color ratio methods [34]–[36] and color difference methods [33], [37]–[40] used correlations between luminance (green channel), and chrominance (red and blue) to create a model to estimate the missing values. However, they are sensitive to the intensity of the spectral correlations among the color channels. Residual interpolation methods [10]–[14] computes for the residuals between an image estimate and the mosaic image to produce better quality images. Frequency based methods transformed the CFA mosaic image to the frequency domain representation and applied filters in order to reconstruct the color image [41]–[43]. Sparse representation methods, which were commonly used on image restoration, have also been applied to image demosaicking. Hua et al. [15], proposed a context-aware dictionary learning algorithm which learn sparse models for different context categories. Mairal et al. [44], [45], used non-local sparse models for image demosacking.

Wu et al. [46] proposed efficient regression prior as a post-processing step to boost the performance of demosaicking methods. Wang [8] used a multilayer neural network that [8] directly estimate the result for demosaicking. However, the output lacks concentrated texture. In contrast, our method uses a totally different architecture to estimate the residuals. Michaël et al. [47] and Tan et al. [48] are concurrent works which also trained deep neural networks for the task. In contrast, we trained multiple models that specializes on different types of image patches. We also offer an intelligent adaptive fusion scheme that can accomodate different textured images in its reconstruction.

III. Proposed Deep Demosaicking Method

We propose a two-step process to perform the demosaicking task. The first step performs a fast interpolation to estimate the missing color values in the mosaic image x outputting an initial image which we refer to as $\hat{x}$. We chose gradient-corrected bilinear interpolation (GCBI) [2] method because of its efficiency and superior performance compared to simple bilinear interpolation, but it still suffers from unwanted color artifacts from incorrectly interpolated values. The second step addresses this issue by training a convolutional neural network, shown in Fig. 2, following the fully convolutional network architecture [20] to correct the color values and reduce the artifacts. The proposed two-step process flowchart is shown in Fig. 3. To further improve the quality, we trained multiple

1057-7149 (c) 2018 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

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/TIP.2018.2803341, IEEE Transactions on Image Processing

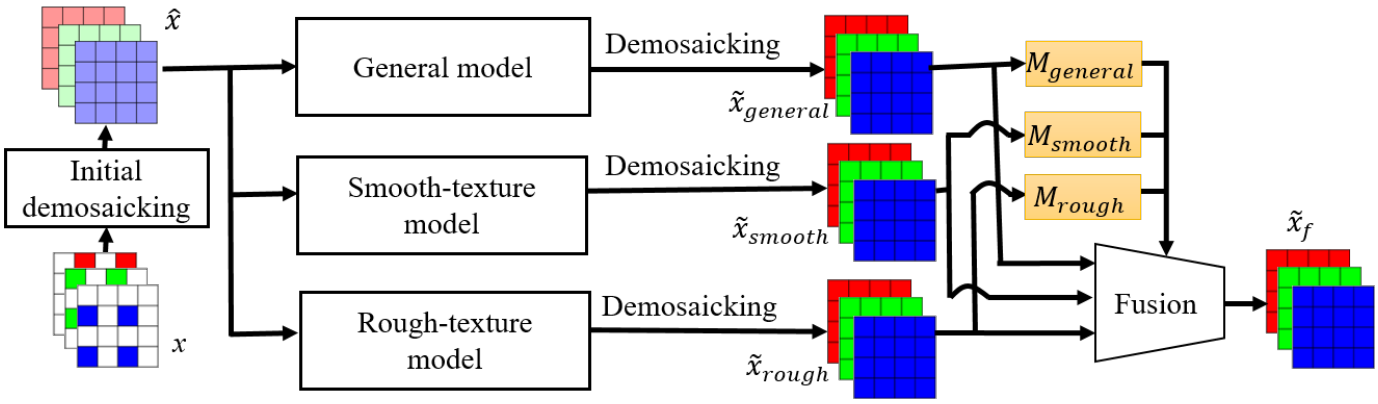


Fig. 4: This image shows the multi-model demosaicking process in which three different models are trained to specialize in a subset of the data. The outputs of these three models are then adaptively fused together using the weighted double interpolation to produce the final image.

models on different subsets of the dataset and adaptively fuse their outputs together. The idea is to train different models specializing in a specific subset of the dataset to make the model robust to different types of inputs.

A. Deep Demosaicking Network Architecture

Our deep demosaicking network mainly consists of convolutional layers and a residual layer at the end. Table I shows the comparison between FCN and our proposed model. We patterned our network to the architecture of FCN [20] but used only a quarter of the number of filters at every layer to speed up the computations. To prevent down-sampling, we removed the pooling layers and added zero-padding before performing the convolutions. The filter sizes of the convolutional layers that originally followed the pooling layers in the FCN architecture were increased to 5×5 to capture the same receptive field as if the pooling layers were in place.

TABLE I: Comparison of model architecture.

FCN [20]		Proposed	
layer	filter(size)	layer	filter(size)
conv1	$64(3 \times 3)$ $64(3 \times 3)$	conv1	$16(3 \times 3)$ $16(3 \times 3)$
max pool		conv2	$32(5 \times 5)$ $32(3 \times 3)$
conv2	$128(3 \times 3)$ $128(3 \times 3)$		
max pool		conv3	$64(5 \times 5)$ $64(3 \times 3)$ $64(3 \times 3)$
conv3	$256(3 \times 3)$ $256(3 \times 3)$ $256(3 \times 3)$		
max pool		conv4	$128(5 \times 5)$ $128(3 \times 3)$ $128(3 \times 3)$
conv4	$512(3 \times 3)$ $512(3 \times 3)$ $512(3 \times 3)$		
max pool		conv5	$128(5 \times 5)$ $128(3 \times 3)$ $128(3 \times 3)$
conv5	$512(3 \times 3)$ $512(3 \times 3)$ $512(3 \times 3)$		
max pool		conv6	$32(5 \times 5)$
conv6	$4096(1 \times 1)$	residual layer	
conv7	$4096(1 \times 1)$		

The last layer, which we call the residual layer, starts with 3 filters sized 1x1 generating a filter map $\hat{r}$, which is the estimated residuals. The filter map $\hat{r}$ will then be added together with the initial image $\hat{x}$ forming a combined image $\tilde{x} = \hat{x} + \hat{r}$. The input mosaic image x would already contain the true color values on certain pixel locations, hence, we replace those pixels from the combined image $\tilde{x}$. This whole process is shown in Fig. 2.

We want to design the network to approximate the residuals $r = y - \hat{x}$, which is the difference between the initial image $\hat{x}$ and the ground truth image y. To do this, we defined the objective function to minimize the Euclidean loss between the actual residuals r and the estimated residuals $\hat{r}$, as shown in Eq. 1, where i refers to the i^{th} image and N refers to the training batch size, $\tilde{x}_i$ is the i^{th} output image in a batch of output images $\tilde{X}$ and $\hat{r}_i$ is the corresponding estimated residual image; y_i is the i^{th} ground truth image in a batch of ground truth images $\tilde{X}$ and r_i is the corresponding actual residual image.

$$\begin{aligned} \mathcal{L}(Y, \tilde{X}) &= \frac{1}{N} \sum_{i=1}^{N} ||y_i - \tilde{x}_i||_2^2 \\ &= \frac{1}{N} \sum_{i=1}^{N} ||(\hat{x}_i + r_i) - (\hat{x}_i + \hat{r}_i)||_2^2 \\ &= \frac{1}{N} \sum_{i=1}^{N} ||r_i - \hat{r}_i||_2^2, \end{aligned} \tag{1}$$

B. Multi-model training

For all machine learning algorithms, the performance depends heavily on how similar the input is to the training data. Much like the motivations for ensemble techniques, the idea behind training multiple models is to create models specializing in different types of data which improves the generalizability of the model and make it more robust to overfitting.

Using the same architecture, three models were trained using three different subsets of the dataset. As shown in Fig. 4, the first model, which we refer to as the general model, uses all

1057-7149 (c) 2018 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

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/TIP.2018.2803341, IEEE Transactions on Image Processing

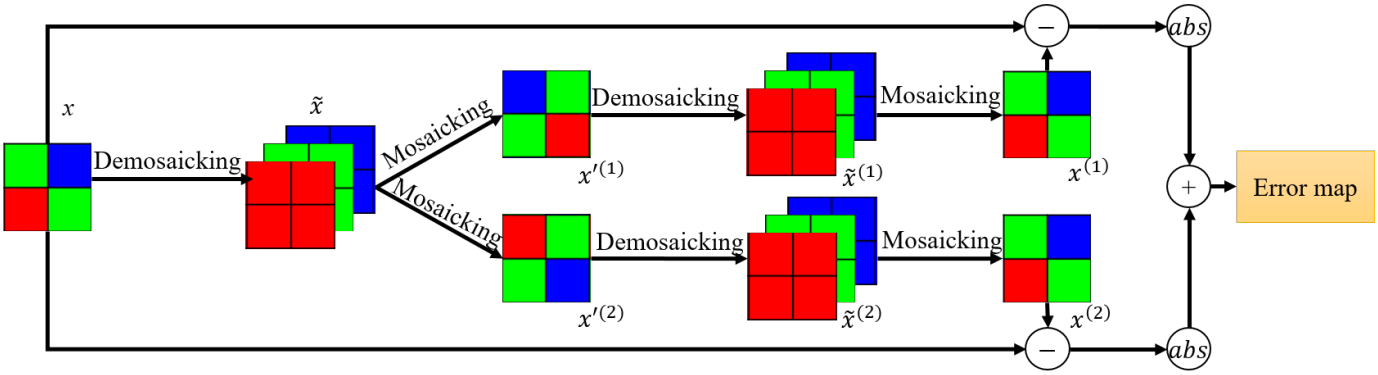


Fig. 6: This image shows the flow of the double interpolation self-validation which is a component of the weighted double interpolation. The goal is to compute for the error map which approximates the error incurred during the demosaicking process.

(a) (b)

Fig. 5: Some example of high-Laplacian energy patch and low-Laplacian energy patch in training data. Image patches showed in (a) have high-Laplacian energy, patches showed in (a) have low-Laplacian energy.

the data while the other two were trained on smooth-textured images and rough-textured images, respectively.

The Laplacian energy $E(I)$ of an image I, is a smoothness metric which can be used to separate smooth-textured images from rough textured images. It is defined as the average of squares of each element Ω of the Laplacian map ΔI. The Laplacian map ΔI is computed as

$$\Delta I = \begin{bmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{bmatrix} * I, \tag{2}$$

$$E(I) = \frac{1}{|\Omega|} \sum_{p \in \Omega} (\Delta I_p)^2, \tag{3}$$

where $*$ is the convolution operator and I_p refers to the image value at pixel position p of image I. Images with rough textures would obtain high Laplacian energy while images with smooth textures would obtain low Laplacian energy. Examples of these are shown in Fig. 5. A threshold t_L on the Laplacian energies must be imposed to clearly define which images are rough and which images are smooth. It is a hyper-parameter that the user can freely choose.

C. Weighted Double Interpolation Fusion

After training, we applied a weighted double interpolation (WDI) as a means to adaptively fuse the outputs of the different models together. WDI is a variant of the double interpolation (DI) self-validation method proposed by [49], [50]. Double Interpolation is a no-reference quality evaluation method. It outputs a score approximating the quality of the demosaicking algorithm on a single image by computing a series of alternating demosaicking and mosaicking as shown in Fig. 6.

First, an input mosaic image x is ran through the DeepDemosaicking model producing a demosaicked image $\tilde{x}$. It will then be mosaicked (down-sampled) to produce two different Bayer pattern-like images $x'^{(1)}$ and $x'^{(2)}$. These two mosaic images will again be demosaicked and mosaicked in a similar manner producing another two mosaic images $x^{(1)}$ and $x^{(2)}$. In a perfect scenario, $x^{(1)}$ and $x^{(2)}$ should be the same as the original mosaic image x. Therefore, for the j^{th} model in the model set $J = \{general, rough, smooth\}$, double interpolation method can generate a demosaicking image $\tilde{x}_j$, and two Bayer pattern image $x_j^{(1)}$ and $x_j^{(2)}$. By comparing $x_j^{(1)}$ and $x_j^{(2)}$ to the original mosaic image x, we can approximate the errors incurred by the j^{th} model. An error map M_j for every model is then computed as shown in Eq. 4. Note that the operations on Eq. 4 are applied on matrices.

$$M_j = \left| x - x_j^{(1)} \right| + \left| x - x_j^{(2)} \right|,$$
$$j \in J, J = \{general, rough, smooth\}, \tag{4}$$

We can use this error map to derive the costs associated with each pixel for the demosaicking model used. The cost for each demosaicking model is computed by applying a bilateral filter on the error map as defined in Eq. 5. This gives different weightings on different pixels according to their spatial and color distances from the original input.

$$Cost_{j,p} = \frac{1}{w_{j,p}} \sum_{q \in \Omega_p} M_{j,q} f\left(\|p - q\|\right) g\left(\|\tilde{x}_{j,p} - \tilde{x}_{j,q}\|\right),$$
$$w_{j,p} = \sum_{q \in \Omega_p} f\left(\|p - q\|\right) g\left(\|\tilde{x}_{j,p} - \tilde{x}_{j,q}\|\right), j \in J, \tag{5}$$

1057-7149 (c) 2018 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

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/TIP.2018.2803341, IEEE Transactions on Image Processing

Fig. 7: The pictures above shows samples of the low resolution (768×512) image dataset release by the Eastman Kodak Company [41].

Fig. 8: The pictures above shows samples of the high resolution images collected by McMaster University for the purpose of color demosaicking [5]. It is refered to as the IMAX dataset in the literatures.

where p and q are pixel positions, Ω_p is the local window centered on a pixel position p, $w_{j,p}$ is a normalization factor for j^{th} model at pixel position p, and $M_{j,q}$ is the error value on the error map M_j at pixel position q. The spatial distance $f(\|p-q\|)$ is a Gaussian on the distance between pixel position p and q, and the value difference $g(\|\tilde{x}_{j,p} - \tilde{x}_{j,q}\|)$ is also a Gaussian of the difference between the image value $\tilde{x}_{j,p} = \{r_{j,p}, g_{j,p}, b_{j,p}\}$ and $\tilde{x}_{j,q} = \{r_{j,q}, g_{j,q}, b_{j,q}\}$, which are the image values at pixel position p and q on the j^{th} model's image result $\tilde{x}_j$. They are computed as follows:

$$f(\|p-q\|) = \frac{1}{\sqrt{2\pi\sigma_s^2}} exp\left(-\frac{\|p-q\|^2}{2\sigma_s^2}\right), \tag{6}$$

$$g(\|\tilde{x}_{j,p} - \tilde{x}_{j,q}\|) = \frac{1}{\sqrt{2\pi\sigma_r^2}} exp\left(-\frac{\|\tilde{x}_{j,p} - \tilde{x}_{j,q}\|^2}{2\sigma_r^2}\right), \tag{7}$$

where σ_s and σ_r in the equation are the standard deviations for the Gaussian distribution, and the color value difference $\|\tilde{x}_{j,p} - \tilde{x}_{j,q}\|$ is computed by l_2-norm $\sqrt{(r_{j,p} - r_{j,q})^2 + (g_{j,p} - g_{j,q})^2 + (b_{j,p} - b_{j,q})^2}$.

Since we want pixels with higher costs to have lower contributions, we used an exponential function to decay the weights proportional to its cost. The equation is shown in below:

$$W_{j,p} = exp\left(\frac{-Cost_{j,p}}{\gamma}\right), j \in J, \tag{8}$$

where γ is a parameter which adjusts the decay rate of the exponential function. $W_{j,p}$ refers to the unnormalized weight value of the j^{th} model for the pixel position p. To normalize the weights, we use a power law normalization, which can further enhance the difference of weights for improving the adaptability, with power n on the converted weight by the following equation:

$$\overline{W}_{j,p} = \frac{(W_{j,p})^n}{\sum_{k \in J} (W_{k,p})^n}, \tag{9}$$

where n is a free parameter in the range $[1, -\infty)$. The final demosaicked image $\tilde{x}_f$ is generated by adding the output of the

1057-7149 (c) 2018 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

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/TIP.2018.2803341, IEEE Transactions on Image Processing

different demosaicking models weighted by $\overline{W}_{j,p}$, as shown below

$$\tilde{x}_{f,p} = \sum_{j \in J} \overline{W}_{j,p} \tilde{x}_{j,p}, \tag{10}$$

where $\tilde{x}_{f,p}$ is the value on the pixel position p of the final resulting image.

IV. Experimental Results

In this section, we describe the implementation details and analyze the quality of the proposed DeepDemosaicking method by comparing it with standard and state-of-the-art demosaicking algorithms.

A. Implementation details

We used Caffe [51] as our framework for implementing the DeepDemosaicking network. For our training data, we gathered 91 images from Yang et al. [52], 500 images from BSDS 500 dataset [53] (size 481×321), 11 images from ARRI dataset [54] (size 2880×1620 downsampled to 1440×810), 136 images form SuperTex136 dataset [55], and 100 images from General-100 [31], totaling to 838 high resolution images with varying image sizes. This is still considered small for training deep convolutional neural networks. To address this, we augmented the dataset by cutting out 64×64 image patches with an overlap of 32 pixels from the dataset. Generating approximately 100,000 image patches that will be mosaicked using the Bayer pattern. These will then be interpolated using the gradient-corrected bilinear interpolation (GCBI) [2] forming the training data for the DeepDemosaicking network.

We tested the DeepDemosaicking model on the Kodak dataset [41] and IMAX dataset [5], both of which were never seen by the model. The Kodak dataset consists of 24 images with high (3072×2048) and low (768×512) resolution versions. In our experiments, we used only the low-resolution versions of the images. The IMAX dataset is provided by [5], where 18 images with size 500×500 are cropped from the original 2310×1814 high resolution images.

Since we want to train three different models on different subsets of the dataset, we first need to determine which subset of images would be used for each model. The general model was trained on all the training dataset, while the smooth-textured model and the rough-textured model was trained on images having low and high Laplacian energy respectively.

The Laplacian threshold determines how we split our dataset into smooth-textured and rough-textured image patches. It is based on the cumulative distribution function (CDF) of the Laplacian energies of the training data. We calibrated this threshold by evaluating the models trained with varying values for the threshold. We first set aside 100 images as our validation set. We then sampled 12,800 image patches from the training set for both the smooth and rough model at different t_L values, and sampled 25,600 image patches for the general model. Fig. 9 shows the overall improvement on the validation set in terms of PSNR with respect to the general model baseline where no dataset splitting was performed. The best performing model on a separate validation set was attained at the threshold of 60th percentile.

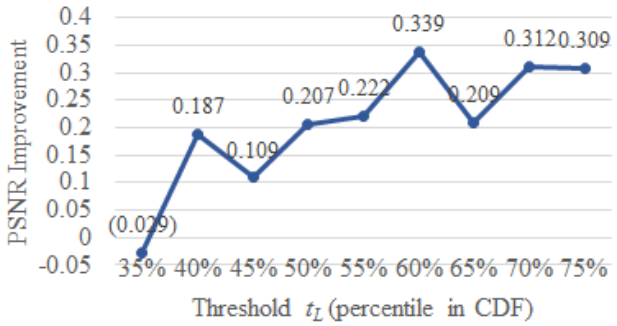


Fig. 9: We evaluated the multi-model scheme with different thresholds t_L trained on a smaller subset of the training data. This image shows the PSNR improvement over the general model baseline without dataset splitting at each value of t_L from 35% to 75%. The value of t_L is based on the CDF of the Laplacian energies of the training data.

All images having a Laplacian energy less than or equal to the threshold t_L were considered smooth-textured, and all images having a Laplacian energy higher than t_L were considered rough-textured. We adaptively fused the models by the applying the weighted double interpolation on the three models with the Gaussian standard deviations $\sigma_s = 4$, $\sigma_r = 0.47$ (the values of these parameters are determined based on the suggestion of [50]), and local window size $\Omega_p = (3 \times 3)$. In addition, we found the optimal parameters $\gamma = 7$ for Eq. 8 and $n = 2$ for Eq. 9 by appling simulated annealing algorithm [56] on the validation dataset.

The DeepDemosaicking network was trained using stochastic gradient descent (SGD) [57] with momentum with the following parameters:

- Base learning rate = 0.1
- Momentum = 0.9
- Weight Decay = 0.0001
- Clip gradients = 0.1
- Batch size = 64
- Training iterations = 160,000 (dividing the learning rate by 10 after 80,000)

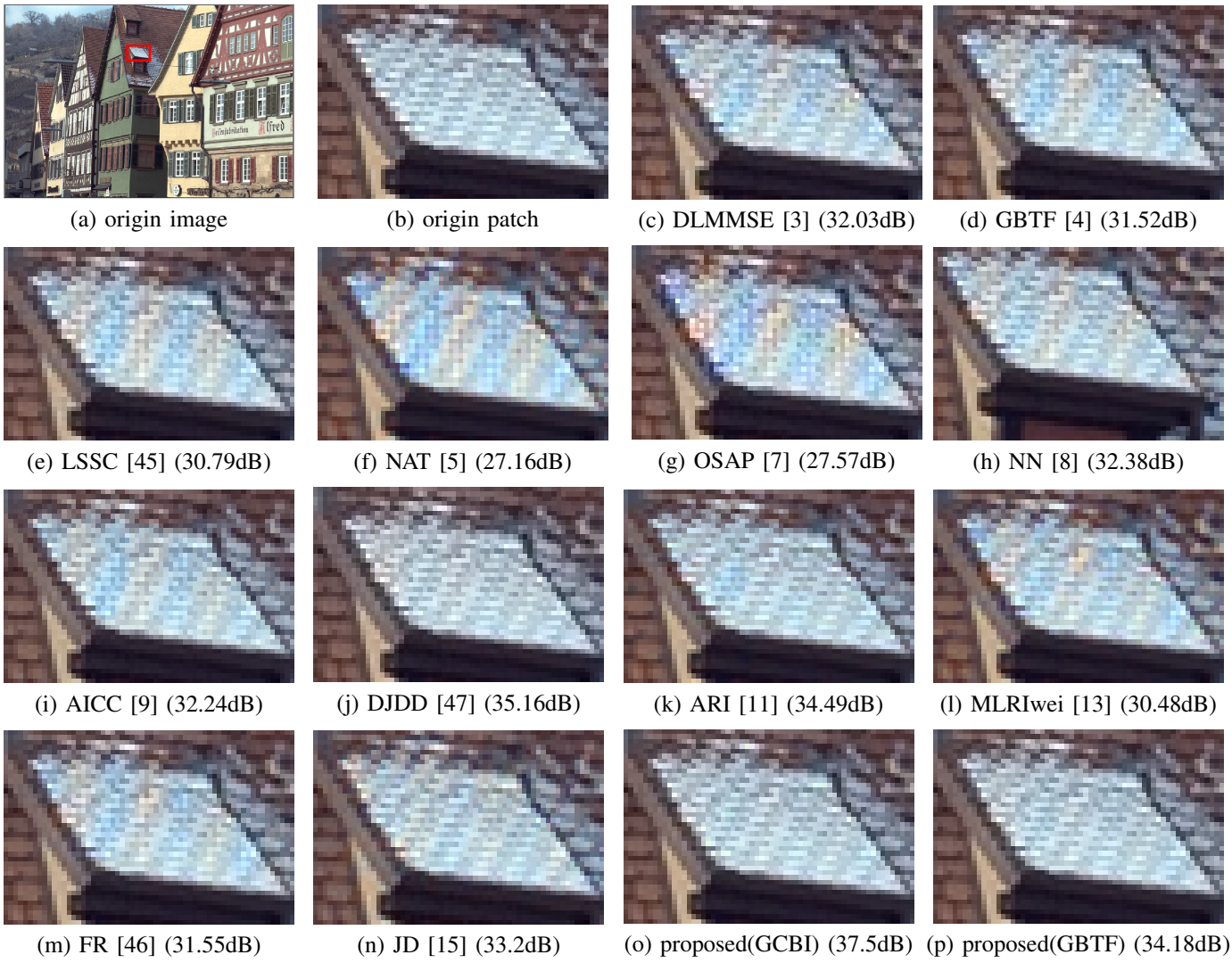
B. Ablation Experiments

To determine the contribution of the components that we proposed, we performed an ablation study. We evaluated the models: (1) single model without the residual layer, (2) single model with residual layer, (3) the multi-model (without the residual layer) with WDI, (4) our proposed multi-model with WDI. Table II shows the performance of the above four settings. We observe a significant performance increase from the model with residual layer as opposed to without. This shows that it is indeed better to model the residuals for the task of image demosaicking. Our proposed multi-model fusion scheme further improves the performance.

We also compare with another baseline model where the multi-model scheme fuses three general models as opposed

1057-7149 (c) 2018 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

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/TIP.2018.2803341, IEEE Transactions on Image Processing



(a) origin image (b) origin patch (c) DLMMSE [3] (32.03dB) (d) GBTF [4] (31.52dB)

(e) LSSC [45] (30.79dB) (f) NAT [5] (27.16dB) (g) OSAP [7] (27.57dB) (h) NN [8] (32.38dB)

(i) AICC [9] (32.24dB) (j) DJDD [47] (35.16dB) (k) ARI [11] (34.49dB) (l) MLRIwei [13] (30.48dB)

(m) FR [46] (31.55dB) (n) JD [15] (33.2dB) (o) proposed(GCBI) (37.5dB) (p) proposed(GBTF) (34.18dB)

Fig. 10: Results of the various demosaicking methods on a small image patch showing a tiled texture of the roof in image(8) of the Kodak dataset [41].

to having general, rough, and smooth models. We randomly shuffled our dataset and split it equally into 3 parts to train three general models. The resulting PSNR and SSIM are shown in Table III. It can be observed that the proposed framework, having models specializing in different types of data, improves the performance by 0.6561 in terms of PSNR.

TABLE II: This table lists the results of our ablation experiments. All methods were initialized using GCBI interpolation and trained using the same parameters.

	Kodak+IMAX	
	PSNR	SSIM
single model (w/o residual)	35.834	0.9815
single model (with residual)	39.950	0.9913
multi-model (w/o residual+WDI)	36.038	0.9822
multi-model (with residual+WDI)	**40.147**	**0.9916**

TABLE III: This table compares the performance of using three general models as opposed to having a combination of general, rough, and smooth models.

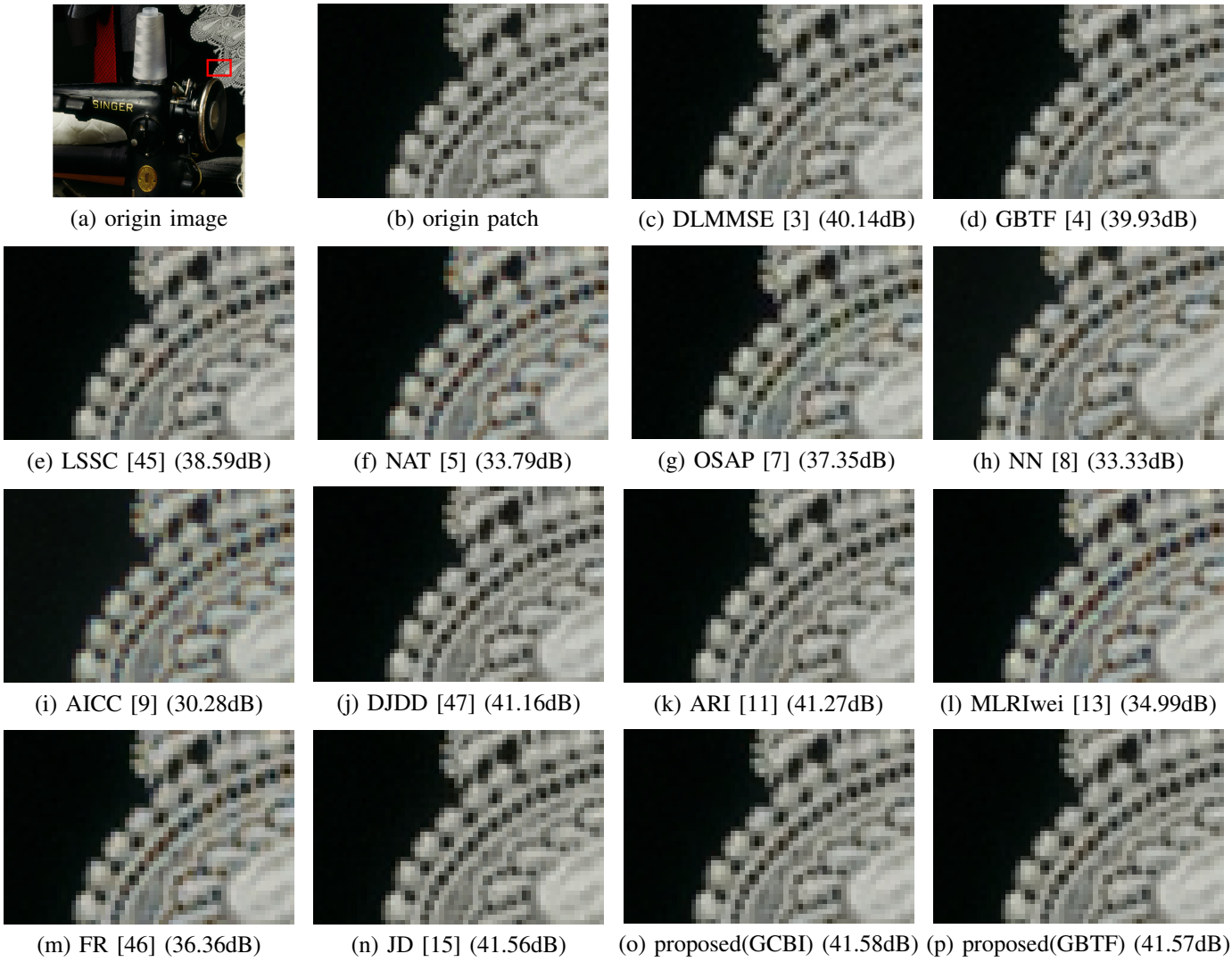
	Kodak+IMAX	
	PSNR	SSIM
Three General Models	39.4909	0.9906
General + Rough + Smooth	**40.147**	**0.9916**

C. Comparison with state-of-the-art algorithms

The results were evaluated using the peak signal-to-noise ratio (PSNR) and structural similarity (SSIM) index. We removed 10 pixels around the borders of the output and the ground truth image to remove the boundary artifacts and keep the evaluations consistent with previous works [7]. Higher PSNR and SSIM values represent higher quality reconstructions. We compared our results with both standard and current state-of-the-art algorithms such as directional linear minimum mean squared error (DLMMSE) [3], gradient based threshold

1057-7149 (c) 2018 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

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/TIP.2018.2803341, IEEE Transactions on Image Processing



(a) origin image (b) origin patch (c) DLMMSE [3] (40.14dB) (d) GBTF [4] (39.93dB)

(e) LSSC [45] (38.59dB) (f) NAT [5] (33.79dB) (g) OSAP [7] (37.35dB) (h) NN [8] (33.33dB)

(i) AICC [9] (30.28dB) (j) DJDD [47] (41.16dB) (k) ARI [11] (41.27dB) (l) MLRIwei [13] (34.99dB)

(m) FR [46] (36.36dB) (n) JD [15] (41.56dB) (o) proposed(GCBI) (41.58dB) (p) proposed(GBTF) (41.57dB)

Fig. 11: Results of the various demosaicking methods on a small image patch showing an intricate texture of a lace in image(8) of the IMAX dataset [5].

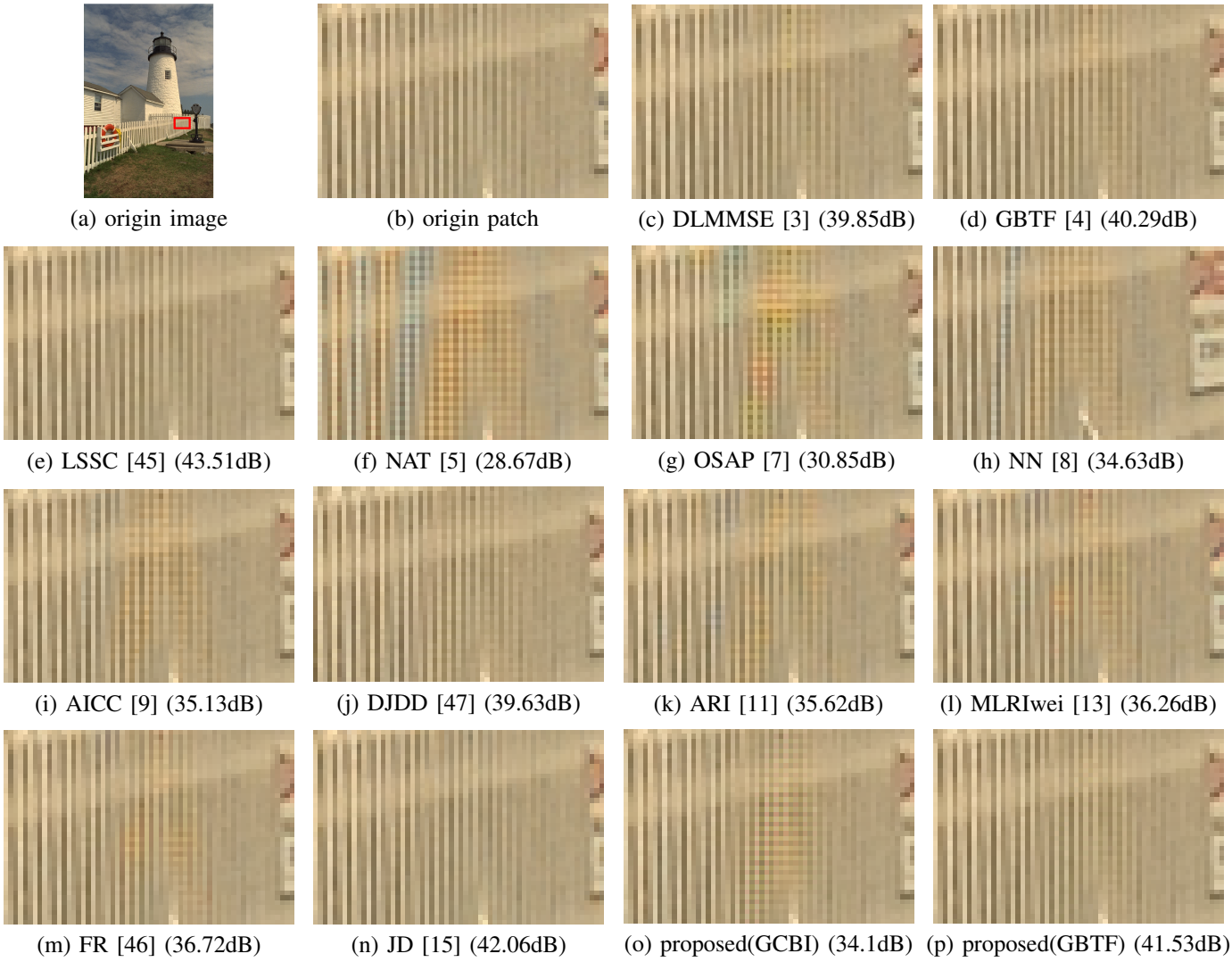
free color filter array interpolation (GBTF) [4], non-local sparse models (LSSC) [45], local directional interpolation and nonlocal adaptive thresholding (NAT) [5], one-step alternating projections (OSAP) [7], multilayer neural network (NN) [8], adaptive inter-channel correlation (AICC) [9], residual interpolation (ARI, MLRIwei) [11], [13], fused regression (FR) [46], deep joint demosaicking and denoising (DJDD) [47], and lastly joint dictionary learning (JD) [15]. We used the results from the respective authors' publicly available code for comparison. The learning based methods were trained with the same data set. Table IV and VI lists the performance of all the methods on the Kodak and IMAX datasets respectively. We can observe that for all the images in the Kodak dataset, DeepDemosaicking performed better than all the other methods mentioned. Since our method starts out by applying GCBI, which performed the worst in most cases, we can see that our DeepDemosaicking network does a good job of enhancing the quality of the interpolation. On the IMAX dataset (Table VI), our method performed comparably in terms of average PSNR and SSIM, performing only slightly less than [11]. These results also show that our method handles both high and low resolution images better than the other methods. Table V and Table VII show a summary of these results.

Aside from using PSNR and SSIM for evaluation, Fig. 10–Fig. 12 shows image results for visual evaluation and comparison of the methods. Fig. 10 shows an image patch of the roof in image 8 of the Kodak dataset. Chromatic aliases can be easily spotted in the results of GCBI [2] (Fig. 10c), DLMMSE [3] (Fig. 10c), GBTF [4] (Fig. 10d), LSSC [45] (Fig. 10e), NAT [5] (Fig. 10f), OSAP [7] (Fig. 10g), NN [8] (Fig. 10h), AICC [9] (Fig. 10i) and MLRIwei [13] (Fig. 10l), while it is almost invisible in the result of DeepDemosaicking (Fig. 10o and Fig. 10p). Fig. 11 shows an example of an image from the IMAX dataset with complex texture. False color artifacts are evident from other results but can hardly be seen in the result of our proposed method.

Fig. 12 shows the fence in image 19 of the Kodak dataset as an example of an image with strong vertical textures. It can be

1057-7149 (c) 2018 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

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/TIP.2018.2803341, IEEE Transactions on Image Processing



(a) origin image (b) origin patch (c) DLMMSE [3] (39.85dB) (d) GBTF [4] (40.29dB)

(e) LSSC [45] (43.51dB) (f) NAT [5] (28.67dB) (g) OSAP [7] (30.85dB) (h) NN [8] (34.63dB)

(i) AICC [9] (35.13dB) (j) DJDD [47] (39.63dB) (k) ARI [11] (35.62dB) (l) MLRIwei [13] (36.26dB)

(m) FR [46] (36.72dB) (n) JD [15] (42.06dB) (o) proposed(GCBI) (34.1dB) (p) proposed(GBTF) (41.53dB)

Fig. 12: Results of the various demosaicking methods on a small image patch showing a fence with strong vertical textures of image(19) in Kodak dataset [41].

observed that most results had chromatic aliases and zippering including our method (Fig. 10o). To investigate this issue, and we found that by changing the initial demosaicking algorithm to a more complex one such as GBTF [4], these artifacts can be eliminated. This shows that our method can also act as a framework for refining the outputs of other demosaicking algorithms.

To evaluate the time complexity of these methods, we estimated their average execution time on the Kodak and IMAX dataset. The authors of the other methods made their implementations public. We used a desktop machine equipped with an Intel Core i7-6700 3.4GHz, 32 GB memory, and an Nvidia GTX-1080 GPU. The results are listed on Table IX. Since most of the previous methods does not leverage on the processing power of GPUs, we included the execution times of both CPU and GPU to give a fair comparison. With the GPU, our method was not the fastest, but among the methods that produced high demosaicking quality, our method was the most prominent with a reasonable running time. Without the GPU, we can observe that our method still performs faster than some of the previous methods such as JD [15], NAT [5], and LSSC [45]. For deep learning approaches, there is a trade off between quality and computational power. However, technology advances fast and newer hardware that specializes on convolution operations are actively being developed so it will be less of an issue in the near future.

V. Conclusion

In this paper, we proposed a novel method for demosaicking images. It first utilizes an efficient interpolation and then enhances its quality based on multiple trained convolutional neural networks and weighted double interpolation fusion scheme. Our model is able to estimate the residuals between low quality interpolated images and the high-quality version of the image. Experimental results show that the proposed method outperforms several existing or state-of-the-art techniques in terms of both subjective and objective evaluations.

1057-7149 (c) 2018 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

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/TIP.2018.2803341, IEEE Transactions on Image Processing

TABLE IV: PSNR and SSIM of 24 Kodak images, the value highlighted in bold type is the highest PSNR or SSIM value of all comparative approach. proposed(GCBI) is the proposed method with initial interpolation by GCBI and proposed(GBTF) is the proposed method with initial interpolation by GBTF.

No.	1		2		3		4		5		6		7		8	
	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
GCBI [2]	32.06	0.9654	37.96	0.9945	39.81	0.9915	38.70	0.9934	33.52	0.9798	33.38	0.9727	39.56	0.9927	29.14	0.9600
DLMMSE [3]	38.52	0.9890	40.92	0.9815	42.76	0.9924	40.58	0.9888	38.11	0.9893	40.27	0.9919	42.40	0.9911	36.08	0.9874
GBTF [4]	39.62	0.9877	41.53	0.9945	43.31	0.9909	40.71	0.9927	38.43	0.9878	41.05	0.9926	42.91	0.9908	37.02	0.9870
LSSC [45]	41.37	0.9954	42.17	0.9984	44.36	0.9967	42.44	0.9973	39.63	0.9949	41.74	0.9957	44.09	0.9967	37.59	0.9914
NAT [5]	34.45	0.9772	39.78	0.9861	41.94	0.9928	39.67	0.9898	35.82	0.9864	35.80	0.9838	41.50	0.9919	32.61	0.9772
OSAP [7]	38.02	0.9905	39.38	0.9963	41.22	0.9957	39.75	0.9956	37.18	0.9915	38.84	0.9922	41.50	0.9953	35.71	0.9880
NN [8]	38.66	0.9911	40.76	0.9984	43.89	0.9968	41.10	0.9968	39.39	0.9943	40.36	0.9940	44.04	0.9970	36.51	0.9888
AICC [9]	40.98	0.9927	40.47	0.9964	43.80	0.9946	41.02	0.9956	38.17	0.9908	41.13	0.9918	42.27	0.9947	37.43	0.9894
ARI [11]	38.81	0.9911	39.66	0.9949	42.68	0.9959	40.70	0.9955	38.27	0.9932	40.52	0.9938	42.67	0.9954	34.98	0.9847
MLRIwei [13]	36.32	0.9856	40.46	0.9950	42.57	0.9959	40.82	0.9953	37.30	0.9917	38.74	0.9920	42.58	0.9956	34.50	0.9844
FR [46]	39.66	0.9928	41.62	0.9976	44.00	0.9964	41.50	0.9967	39.27	0.9940	41.24	0.9949	43.75	0.9962	37.34	0.9907
JD [15]	36.12	0.9681	40.21	0.9848	42.26	0.9767	41.70	0.9816	36.54	0.9718	38.80	0.9780	41.81	0.9786	33.87	0.9658
DJDD [47]	41.60	0.9955	41.70	0.9979	43.80	0.9967	42.30	0.9975	39.40	0.9947	41.70	0.9956	44.10	0.9968	37.70	0.9917
proposed(GCBI)	41.96	0.9958	42.14	0.9976	**44.83**	0.9968	42.85	0.9976	**40.56**	0.9956	42.46	0.9960	44.90	0.9970	38.11	0.9926
proposed(GBTF)	**42.07**	**0.9962**	**42.31**	**0.9985**	44.81	**0.9973**	**43.04**	**0.9977**	40.42	**0.9957**	**42.60**	**0.9964**	**44.91**	**0.9973**	**38.47**	**0.9928**
No.	9		10		11		12		13		14		15		16	
	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
GCBI [2]	38.07	0.9783	38.59	0.9823	34.77	0.9776	38.75	0.9885	29.62	0.9638	34.62	0.9764	38.05	0.9860	36.52	0.9691
DLMMSE [3]	43.15	0.9893	42.53	0.9883	40.10	0.9904	43.53	0.9937	34.82	0.9860	37.05	0.9908	39.88	0.9901	43.84	0.9899
GBTF [4]	43.63	0.9856	42.94	0.9870	40.90	0.9881	44.15	0.9938	35.50	0.9858	37.24	0.9873	40.03	0.9913	44.49	0.9870
LSSC [45]	43.86	0.9898	43.22	0.9919	41.57	0.9932	44.92	0.9969	36.57	0.9915	38.98	0.9933	41.74	0.9944	44.89	0.9943
NAT [5]	40.53	0.9846	40.84	0.9861	36.95	0.9850	41.02	0.9914	30.74	0.9706	36.32	0.9844	38.49	0.9886	39.04	0.9814
OSAP [7]	41.75	0.9888	41.80	0.9905	39.28	0.9909	42.54	0.9954	34.51	0.9872	35.49	0.9897	39.02	0.9921	42.24	0.9909
NN [8]	43.32	0.9887	42.96	0.9910	40.30	0.9919	43.62	0.9959	35.38	0.9887	38.78	0.9918	40.56	0.9930	43.57	0.9923
AICC [9]	43.38	0.9888	42.53	0.9904	40.42	0.9912	44.18	0.9957	36.50	0.9905	37.80	0.9892	40.20	0.9913	44.42	0.9903
ARI [11]	41.57	0.9785	41.55	0.9888	39.63	0.9904	43.17	0.9956	35.21	0.9877	37.55	0.9918	38.61	0.9893	43.25	0.9930
MLRIwei [13]	41.82	0.9853	41.92	0.9894	38.85	0.9907	42.94	0.9952	32.53	0.9802	37.44	0.9896	39.14	0.9916	42.46	0.9915
FR [46]	43.65	0.9898	43.28	0.9916	41.16	0.9934	44.58	0.9965	35.75	0.9896	38.27	0.9926	40.28	0.9938	44.60	0.9936
JD [15]	41.19	0.9661	41.33	0.9699	37.98	0.9697	42.07	0.9826	35.07	0.9725	35.91	0.9653	40.19	0.9801	41.62	0.9676
DJDD [47]	43.30	0.9891	43.50	0.9917	41.50	0.9938	44.50	0.9967	37.50	0.9929	39.20	0.9936	41.50	0.9940	44.70	0.9940
proposed(GCBI)	44.09	**0.9901**	43.75	0.9923	42.23	0.9941	**45.26**	0.9970	37.67	0.9935	40.27	0.9944	**42.09**	0.9945	45.44	0.9946
proposed(GBTF)	**44.11**	0.9900	**43.76**	**0.9924**	**42.37**	**0.9947**	45.26	**0.9972**	**37.89**	**0.9936**	**40.30**	**0.9950**	42.03	**0.9949**	**45.56**	**0.9951**
No.	17		18		19		20		21		22		23		24	
	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
GCBI [2]	37.70	0.9823	33.91	0.9763	33.67	0.9807	37.32	0.9808	34.15	0.9798	35.43	0.9831	41.00	0.9939	32.22	0.9749
DLMMSE [3]	41.80	0.9909	37.53	0.9720	41.05	0.9772	41.27	0.9803	39.18	0.9760	38.47	0.9783	43.31	0.9759	35.50	0.9793
GBTF [4]	42.10	0.9881	37.85	0.9877	41.83	0.9863	41.83	0.9866	39.78	0.9896	38.86	0.9835	43.66	0.9932	35.60	0.9830
LSSC [45]	42.16	0.9915	38.45	0.9887	42.31	0.9928	42.27	**0.9909**	40.69	0.9934	39.32	0.9911	44.38	0.9965	35.97	0.9889
NAT [5]	39.06	0.9864	34.83	0.9669	37.51	0.9738	39.41	0.9788	35.92	0.9721	37.28	0.9767	42.28	0.9773	33.35	0.9735
OSAP [7]	41.29	0.9909	37.16	0.9812	40.01	0.9867	40.51	0.9854	38.83	0.9867	37.62	0.9854	41.62	0.9896	34.67	0.9835
NN [8]	41.81	0.9905	37.76	0.9872	40.95	0.9920	41.97	0.9904	39.72	0.9921	39.03	0.9911	44.34	**0.9968**	35.70	0.9886
AICC [9]	41.87	0.9902	37.31	0.9861	41.59	0.9918	41.69	0.9869	40.41	0.9915	38.26	0.9876	43.07	0.9951	35.84	0.9876
ARI [11]	41.14	0.9900	36.87	0.9833	40.46	0.9902	40.86	0.9867	39.24	0.9911	38.06	0.9878	43.17	0.9960	35.35	0.9872
MLRIwei [13]	40.67	0.9896	35.89	0.9834	39.43	0.9899	40.29	0.9879	37.71	0.9899	38.24	0.9892	43.09	0.9962	34.41	0.9855
FR [46]	42.29	0.9917	38.15	0.9865	41.85	0.9884	42.32	0.9887	40.31	0.9895	39.48	0.9885	44.08	0.9943	35.65	0.9868
JD [15]	41.03	0.9673	36.96	0.9745	39.33	0.9667	40.21	0.9679	38.17	0.9732	37.46	0.9685	43.39	0.9796	34.39	0.9623
DJDD [47]	42.00	0.9909	**39.30**	0.9890	42.20	**0.9935**	42.30	0.9904	41.00	0.9937	39.30	0.9918	44.10	0.9963	36.60	0.9900
proposed(GCBI)	**42.76**	0.9917	39.05	0.9895	42.56	0.9933	**42.97**	0.9908	**41.51**	0.9940	39.96	0.9920	44.97	0.9965	36.58	0.9905
proposed(GBTF)	42.71	**0.9919**	39.26	**0.9899**	**42.88**	**0.9935**	42.91	**0.9909**	41.35	**0.9942**	**39.99**	**0.9922**	**45.13**	**0.9968**	**36.80**	**0.9909**

TABLE V: Average PSNR and SSIM of 24 images in Kodak dataset, the value highlighted in bold type is the highest PSNR or SSIM value of all comparative approach. proposed(GCBI) is the proposed method with initial interpolation by GCBI and proposed(GBTF) is the proposed method with initial interpolation by GBTF.

	DLMMSE [3]	GBTF [4]	LSSC [45]	NAT [5]	OSAP [7]	NN [8]	AICC [9]
PSNR	40.110	40.623	41.445	37.714	39.165	40.603	40.614
SSIM	0.9858	0.9887	0.9936	0.9818	0.9900	0.9925	0.9913
	DJDD [47]	ARI [11]	MLRIwei [13]	FR [46]	JD [15]	proposed(GCBI)	**proposed(GBTF)**
PSNR	41.450	39.749	39.171	41.002	39.066	42.041	**42.122**
SSIM	0.9937	0.9905	0.9900	0.9923	0.9725	0.9941	**0.9944**

ACKNOWLEDGEMENT

This work was supported by Ministry of Science and Technology of Taiwan under Grants MOST106-3114-E-011-004, MOST106-2218-E-011-009-MY2, MOST106-2221-E-011-154-MY2.

REFERENCES

[1] B. Bayer, "Color imaging array," Patent US 3 971 065, 1976.

[2] H. S. Malvar, L. He, and R. Cutler, "High-quality linear interpolation for demosaicing of bayer-patterned color images," in *IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP, May 17-21*, 2004, pp. 485–488.

[3] L. Zhang and X. Wu, "Color demosaicking via directional linear minimum mean square-error estimation," *IEEE Trans. Image Processing*, vol. 14, no. 12, pp. 2167–2178, 2005.

[4] I. Pekkucuksen and Y. Altunbasak, "Gradient based threshold free color filter array interpolation," in *Proceedings of the International Conference on Image Processing, ICIP, September 26-29*, 2010, pp. 137–140.

[5] L. Zhang, X. Wu, A. Buades, and X. Li, "Color demosaicking by local directional interpolation and nonlocal adaptive thresholding," *J. Electronic Imaging*, vol. 20, no. 2, p. 023016, 2011.

[6] F. He, Y. F. Wang, and K. Hua, "Self-learning approach to color demosaicking via support vector regression," in *19th IEEE International Conference on Image Processing, ICIP, September 30 - October 3*, 2012, pp. 2765–2768.

[7] Y. M. Lu, M. Karzand, and M. Vetterli, "Demosaicking by alternating projections: Theory and fast one-step implementation," *IEEE Trans. Image Processing*, vol. 19, no. 8, pp. 2085–2098, 2010.

[8] Y. Wang, "A multilayer neural network for image demosaicking," in

1057-7149 (c) 2018 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

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/TIP.2018.2803341, IEEE Transactions on Image Processing

TABLE VI: PSNR and SSIM of 18 IMAX images, the value highlighted in bold type is the highest PSNR or SSIM value of all comparative approach. proposed(GCBI) is the proposed method with initial interpolation by GCBI and proposed(GBTF) is the proposed method with initial interpolation by GBTF.

No.	1		2		3		4		5		6		7		8		9	
	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
GCBI [2]	27.41	0.9640	36.91	0.9917	37.86	0.9895	36.17	0.9952	38.67	0.9956	37.09	0.9863	37.58	0.9909	31.57	0.9827	31.30	0.9771
DLMMSE [3]	26.98	0.9321	36.34	0.9631	37.24	0.9598	36.61	0.9600	38.80	0.9588	37.24	0.9541	37.27	0.9597	30.45	0.9484	29.31	0.9373
GBTF [4]	26.89	0.9602	36.25	0.9892	37.04	0.9858	36.65	0.9936	38.65	0.9937	37.10	0.9851	37.09	0.9897	30.08	0.9754	29.02	0.9601
LSSC [45]	28.31	0.9705	38.19	0.9939	39.39	0.9923	38.03	0.9969	40.07	0.9968	38.31	0.9895	38.70	0.9932	32.23	0.9850	32.53	0.9824
NAT [5]	29.18	0.9466	38.78	0.9688	39.57	0.9667	38.88	0.9652	40.74	0.9643	38.70	0.9599	38.95	0.9651	33.84	0.9603	32.88	0.9572
OSAP [7]	25.24	0.9375	34.67	0.9790	35.61	0.9734	35.37	0.9838	37.07	0.9826	36.01	0.9731	36.11	0.9792	28.60	0.9580	27.76	0.9400
NN [8]	27.05	0.9589	30.03	0.9575	29.43	0.9471	32.21	0.9862	35.48	0.9921	31.68	0.9568	30.27	0.9483	28.34	0.9560	28.87	0.9584
AICC [9]	26.88	0.9567	29.73	0.9550	29.21	0.9442	31.77	0.9849	35.11	0.9914	31.44	0.9554	30.07	0.9471	27.90	0.9534	28.32	0.9525
ARI [11]	29.63	**0.9763**	39.25	0.9942	40.19	0.9936	40.03	0.9975	40.60	0.9969	39.02	0.9895	39.38	0.9934	35.63	**0.9916**	34.69	0.9879
MLRIwei [13]	29.41	0.9760	39.09	0.9938	40.17	0.9938	39.81	0.9975	40.62	0.9970	39.08	0.9896	39.22	0.9933	35.41	0.9915	33.18	0.9850
FR [46]	**29.80**	0.9749	39.62	0.9919	40.41	0.9913	**40.14**	0.9939	41.23	0.9929	**39.37**	0.9865	**39.65**	0.9904	35.33	0.9876	34.21	0.9842
JD [15]	29.03	0.9586	38.36	0.9804	39.41	0.9814	38.58	0.9786	40.05	0.9755	38.71	0.9741	39.12	0.9784	34.16	0.9783	34.10	0.9775
DJDD [47]	28.70	0.9741	34.80	0.9767	34.20	0.9793	37.00	0.9883	33.90	0.9800	38.60	0.9823	40.50	0.9852	**40.00**	0.9817	**38.20**	**0.9930**
proposed(GCBI)	29.26	0.9757	**39.66**	**0.9949**	**40.85**	**0.9944**	39.39	**0.9976**	**41.50**	**0.9975**	39.10	**0.9904**	39.48	**0.9943**	34.98	0.9901	34.30	0.9874
proposed(GBTF)	28.94	0.9735	39.20	0.9943	40.36	0.9936	39.15	0.9974	41.27	0.9973	38.97	0.9900	39.14	0.9939	34.49	0.9888	33.34	0.9840
No.	10		11		12		13		14		15		16		17		18	
	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
GCBI [2]	33.15	0.9801	33.20	0.9699	31.26	0.9677	33.87	0.9823	32.01	0.9710	35.17	0.9716	35.02	0.9587	36.15	0.9711	35.28	0.9890
DLMMSE [3]	33.90	0.9540	33.69	0.9437	32.59	0.9646	34.33	0.9813	31.28	0.9447	33.83	0.9392	38.65	0.9759	37.46	0.9725	34.41	0.9601
GBTF [4]	34.04	0.9798	33.62	0.9695	32.69	0.9669	34.74	0.9736	30.92	0.9582	33.21	0.9536	38.97	0.9773	37.50	0.9756	34.31	0.9856
LSSC [45]	34.82	0.9836	34.36	0.9756	33.10	0.9766	37.12	0.9886	32.41	0.9747	37.86	0.9820	39.52	0.9824	38.64	0.9819	37.07	0.9921
NAT [5]	35.10	0.9593	35.04	0.9520	32.60	0.9668	35.94	0.9848	34.09	0.9578	37.83	0.9592	35.96	0.9620	37.50	0.9733	37.00	0.9674
OSAP [7]	32.35	0.9656	32.22	0.9528	31.56	0.9656	33.49	0.9800	29.56	0.9477	31.41	0.9330	37.86	0.9764	36.79	0.9727	33.16	0.9739
NN [8]	31.60	0.9528	30.22	0.9367	32.11	0.9547	36.64	0.9884	32.18	0.9639	34.08	0.9623	33.37	0.9333	31.23	0.7859	31.54	0.9681
AICC [9]	30.75	0.9487	29.85	0.9336	30.05	0.9432	32.73	0.9808	31.56	0.9607	33.22	0.9575	32.18	0.9128	30.62	0.7793	30.92	0.9649
ARI [11]	36.42	0.9884	35.22	0.9779	34.64	0.9805	37.90	0.9899	35.43	0.9827	**39.75**	0.9857	39.71	0.9823	39.51	0.9814	37.88	0.9929
MLRIwei [13]	36.40	0.9885	35.34	0.9775	34.04	0.9799	37.98	0.9904	34.42	0.9805	38.80	0.9831	37.03	0.9706	37.29	0.9789	36.82	0.9918
FR [46]	36.28	0.9840	35.61	0.9762	34.48	0.9669	39.13	0.9708	35.08	0.9765	39.46	0.9803	36.82	0.9633	39.44	0.9772	38.02	0.9907
JD [15]	35.40	0.9727	34.47	0.9621	32.88	0.9439	36.55	0.9434	33.88	0.9544	38.35	0.9589	39.64	0.9621	38.22	0.9603	36.67	0.9756
DJDD [47]	**38.60**	**0.9941**	**40.00**	**0.9931**	**38.50**	**0.9972**	**40.80**	**0.9971**	**38.70**	**0.9897**	39.10	**0.9937**	34.30	**0.9885**	33.00	**0.9837**	35.80	0.9848
proposed(GCBI)	35.68	0.9860	34.95	0.9774	34.95	0.9818	38.70	0.9906	35.11	0.9825	39.40	0.9850	**40.82**	0.9857	**40.40**	0.9832	**38.64**	**0.9939**
proposed(GBTF)	35.68	0.9857	34.82	0.9766	34.74	0.9811	38.45	0.9904	34.81	0.9812	38.57	0.9819	40.76	0.9855	40.34	0.9826	38.23	0.9932

TABLE VII: Average PSNR and SSIM of 18 images in IMAX dataset, the value highlighted in bold type is the highest PSNR or SSIM value of all comparative approach. proposed(GCBI) is the proposed method with initial interpolation by GCBI and proposed(GBTF) is the proposed method is the proposed method with initial interpolation by GBTF.

	DLMMSE [3]	GBTF [4]	LSSC [45]	NAT [5]	OSAP [7]	NN [8]	AICC [9]
PSNR	34.466	34.376	36.148	36.254	33.047	31.464	30.684
SSIM	0.9561	0.9763	0.9854	0.9631	0.9652	0.9504	0.9457
	DJDD [47]	ARI [11]	MLRIwei [13]	FR [46]	JD [15]	**proposed(GCBI)**	proposed(GBTF)
PSNR	36.927	37.494	36.894	37.449	36.532	**37.621**	37.293
SSIM	0.9868	0.9879	0.9866	0.9822	0.9676	**0.9882**	0.9873

TABLE VIII: Average PSNR and SSIM of 42 images in both dataset, the value highlighted in bold type is the highest PSNR or SSIM value of all comparative approach. proposed(GCBI) is the proposed method with initial interpolation by GCBI and proposed(GBTF) is the proposed method is the proposed method with initial interpolation by GBTF.

	DLMMSE [3]	GBTF [4]	LSSC [45]	NAT [5]	OSAP [7]	NN [8]	AICC [9]
PSNR	37.691	37.946	39.175	37.088	36.543	36.686	36.358
SSIM	0.9731	0.9834	0.9901	0.9738	0.9794	0.9744	0.9717
	DJDD [47]	ARI [11]	MLRIwei [13]	FR [46]	JD [15]	**proposed(GCBI)**	proposed(GBTF)
PSNR	39.512	38.783	38.195	39.357	39.480	**40.147**	40.052
SSIM	0.9907	0.9894	0.9886	0.9864	0.9704	**0.9913**	0.9913

IEEE International Conference on Image Processing, ICIP, October 27-30, 2014, pp. 1852–1856.

[9] J. Duran and A. Buades, "A demosaicking algorithm with adaptive inter-channel correlation," *IPOL Journal*, vol. 5, pp. 311–327, 2015.

[10] D. Kiku, Y. Monno, M. Tanaka, and M. Okutomi, "Residual interpolation for color image demosaicking," in *IEEE International Conference on Image Processing, ICIP, September 15-18*, 2013, pp. 2304–2308.

[11] Y. Monno, D. Kiku, M. Tanaka, and M. Okutomi, "Adaptive residual interpolation for color image demosaicking," in *IEEE International Conference on Image Processing, ICIP, September 27-30*, 2015, pp. 3861–3865.

[12] D. Kiku, Y. Monno, M. Tanaka, and M. Okutomi, "Minimized-laplacian residual interpolation for color image demosaicking," in *Digital Photography X, part of the IS&T-SPIE Electronic Imaging Symposium, San Francisco, California, USA, February 2, Proceedings.*, 2014, p. 90230L.

[13] ——, "Beyond color difference: Residual interpolation for color image demosaicking," *IEEE Trans. Image Processing*, vol. 25, no. 3, pp. 1288–1300, 2016.

[14] W. Ye and K. Ma, "Color image demosaicing using iterative residual interpolation," *IEEE Trans. Image Processing*, vol. 24, no. 12, pp. 5879–5891, 2015.

[15] K. Hua, S. C. Hidayati, F. He, C. Wei, and Y. F. Wang, "Context-aware joint dictionary learning for color image demosaicking," *J. Visual Communication and Image Representation*, vol. 38, pp. 230–245, 2016.

[16] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems 25: 26th Annual Conference on Neural Information Processing Systems. Proceedings of a meeting held December 3-6, Lake Tahoe, Nevada, United States.*, 2012, pp. 1106–1114.

[17] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[18] R. Zhang, P. Isola, and A. A. Efros, "Colorful image colorization," in *Computer Vision - ECCV - 14th European Conference, Amsterdam, The Netherlands, October 11-14*, 2016, pp. 649–666.

[19] S. Iizuka, E. Simo-Serra, and H. Ishikawa, "Let there be color!: joint end-to-end learning of global and local image priors for automatic image colorization with simultaneous classification," *ACM Trans. Graph.*, vol. 35, no. 4, p. 110, 2016.

[20] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for

1057-7149 (c) 2018 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

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/TIP.2018.2803341, IEEE Transactions on Image Processing

TABLE IX: Average execution time(second) per image.

	GCBI [2]	DLMMSE [3]	GBTF [4]	LSSC [45]	NAT [5]	OSAP [7]	NN [8]	AICC [9])
Time(s)	0.105	5.022	4.954	93.977	356.986	0.029	0.05	22.332
	ARI [11]	MLRIwei [13]	FR [46]	JD [15]	DJDD [47] (GPU)	DJDD [47] (CPU)	Proposed (GPU)	Proposed (CPU)
Time(s)	15.422	2.090	9.839	461.4	2.936	56.8985	4.186	78.583

semantic segmentation," in *IEEE Conference on Computer Vision and Pattern Recognition, CVPR, June 7-12*, 2015, pp. 3431–3440.

[21] H. Noh, S. Hong, and B. Han, "Learning deconvolution network for semantic segmentation," in *2015 IEEE International Conference on Computer Vision, ICCV, December 7-13*, 2015, pp. 1520–1528.

[22] S. Xie and Z. Tu, "Holistically-nested edge detection," in *2015 IEEE International Conference on Computer Vision, ICCV, December 7-13*, 2015, pp. 1395–1403.

[23] W. Shen, X. Wang, Y. Wang, X. Bai, and Z. Zhang, "Deepcontour: A deep convolutional feature learned by positive-sharing loss for contour detection," in *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2015, Boston, MA, USA, June 7-12, 2015*, 2015, pp. 3982–3991.

[24] W. Shi, J. Caballero, F. Huszár, J. Totz, A. P. Aitken, R. Bishop, D. Rueckert, and Z. Wang, "Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 1874–1883.

[25] K. Zhang, W. Zuo, Y. Chen, D. Meng, and L. Zhang, "Beyond a gaussian denoiser: Residual learning of deep cnn for image denoising," *IEEE Transactions on Image Processing*, 2017.

[26] K.-L. Hua, H. Che-Hao, S. C. Hidayati, W.-H. Cheng, and Y.-J. Chen, "Computer-aided classification of lung nodule on computed tomography images via deep learning technique," *OncoTargets and Therapy*, vol. 8, pp. 2015–2022, Aug 2015.

[27] K. A. Jangtjik, M.-C. Yeh, and K.-L. Hua, "Artist-based classification via deep learning with multi-scale weighted pooling," in *Proceedings of the 24th ACM International Conference on Multimedia*, 2016, pp. 635–639.

[28] J. Sanchez-Riera, K. Srinivasan, K.-L. Hua, W.-H. Cheng, M. A. Hossain, and M. F. Alhamid, "Robust rgb-d hand tracking using deep learning priors," *IEEE Transactions on Circuits and Systems for Video Technology*, 2018.

[29] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *CoRR*, vol. abs/1409.1556, 2014.

[30] C. Dong, C. C. Loy, K. He, and X. Tang, "Learning a deep convolutional network for image super-resolution," in *Computer Vision - ECCV - 13th European Conference, Zurich, Switzerland, September 6-12*, 2014, pp. 184–199.

[31] C. Dong, C. C. Loy, and X. Tang, "Accelerating the super-resolution convolutional neural network," in *Computer Vision - ECCV 2016 - 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part II*, 2016, pp. 391–407.

[32] J. Kim, J. K. Lee, and K. M. Lee, "Accurate image super-resolution using very deep convolutional networks," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[33] J. E. A. Jr. and J. F. H. Jr., "Adaptive color plane interpolation in single sensor color electronic camera," Patent US 5 652 621, 1997.

[34] D. R. Cok, "Signal processing method and apparatus for producing interpolated chrominance values in a sampled color image signal," Patent US 4 642 678, 1987.

[35] R. Lukac and K. N. Plataniotis, "A normalized model for color-ratio based demosaicking schemes," in *Proceedings of the 2004 International Conference on Image Processing, ICIP, October 24-27*, 2004, pp. 1657–1660.

[36] R. Kimmel, "Demosaicing: image reconstruction from color CCD samples," *IEEE Trans. Image Processing*, vol. 8, no. 9, pp. 1221–1228, 1999.

[37] S.-C. Pei and I.-K. Tam, "Effective color interpolation in ccd color filter arrays using signal correlation," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 13, no. 6, pp. 503–513, June 2003.

[38] C. Su, "Highly effective iterative demosaicing using weighted-edge and color-difference interpolations," *IEEE Trans. Consumer Electronics*, vol. 52, no. 2, pp. 639–645, 2006.

[39] K. H. Chung and Y. H. Chan, "Enhanced integrated gradient and its applications to color demosaicing," in *2012 IEEE International Conference on Signal Processing, Communication and Computing (ICSPCC 2012)*, Aug 2012, pp. 378–383.

[40] K. Hirakawa and T. W. Parks, "Adaptive homogeneity-directed demosaicing algorithm," *IEEE Trans. Image Processing*, vol. 14, no. 3, pp. 360–369, 2005.

[41] E. Dubois, "Frequency-domain methods for demosaicking of bayer-sampled color images," *IEEE Signal Processing Letters*, vol. 12, no. 12, pp. 847–850, Dec 2005.

[42] D. Alleysson, S. Süsstrunk, and J. Hérault, "Linear demosaicing inspired by the human visual system," *IEEE Trans. Image Processing*, vol. 14, no. 4, pp. 439–449, 2005.

[43] E. Dubois, "Filter design for adaptive frequency-domain bayer demosaicking," in *Proceedings of the International Conference on Image Processing, ICIP, October 8-11*, 2006, pp. 2705–2708.

[44] J. Mairal, M. Elad, and G. Sapiro, "Sparse representation for color image restoration," *IEEE Trans. Image Processing*, vol. 17, no. 1, pp. 53–69, 2008.

[45] J. Mairal, F. R. Bach, J. Ponce, G. Sapiro, and A. Zisserman, "Non-local sparse models for image restoration," in *IEEE 12th International Conference on Computer Vision, ICCV, September 27 - October 4*, 2009, pp. 2272–2279.

[46] J. Wu, R. Timofte, and L. J. V. Gool, "Demosaicing based on directional difference regression and efficient regression priors," *IEEE Trans. Image Processing*, vol. 25, no. 8, pp. 3862–3874, 2016.

[47] M. Gharbi, G. Chaurasia, S. Paris, and F. Durand, "Deep joint demosaicking and denoising," *ACM Trans. Graph.*, vol. 35, no. 6, pp. 191–191, 2016.

[48] R. Tan, K. Zhang, W. Zuo, and L. Zhang, "Color image demosaicking via deep residual learning," in *Proceedings of the 2017 IEEE International Conference on Multimedia and Expo, ICME*, 2017.

[49] Y.-N. Liu, Y.-C. Lin, and S. Y. Chien, "A no -reference quality evaluation method for cfa demosaicking," in *2010 Digest of Technical Papers International Conference on Consumer Electronics (ICCE)*, Jan 2010.

[50] T. Wang, Y. Liu, and S. Chien, "Color filter array demosaicking using self-validation framework," in *Proceedings of the 2012 IEEE International Conference on Multimedia and Expo, ICME*, 2012, pp. 604–609.

[51] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. B. Girshick, S. Guadarrama, and T. Darrell, "Caffe: Convolutional architecture for fast feature embedding," in *Proceedings of the ACM International Conference on Multimedia, MM, November 03 - 07*, 2014, pp. 675–678.

[52] J. Yang, J. Wright, T. S. Huang, and Y. Ma, "Image super-resolution via sparse representation," *IEEE Trans. Image Processing*, vol. 19, no. 11, pp. 2861–2873, 2010.

[53] Arbelaez, Pablo, M. Maire, C. Fowlkes, and J. Malik, "Contour detection and hierarchical image segmentation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 33, no. 5, pp. 898–916, 2011.

[54] S. Andriani, H. Brendel, T. Seybold, and J. Goldstone, "Beyond the kodak image set: A new reference set of color image sequences," in *IEEE International Conference on Image Processing, ICIP, Melbourne, Australia, September 15-18*, 2013, pp. 2289–2293.

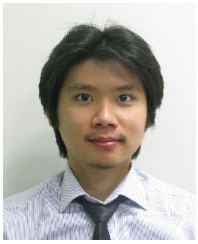
[55] D. Dai, R. Timofte, and L. J. V. Gool, "Jointly optimized regressors for image super-resolution," *Comput. Graph. Forum*, vol. 34, no. 2, pp. 95–104, 2015.

[56] S. Kirkpatrick, C. D. Gelatt, M. P. Vecchi *et al.*, "Optimization by simulated annealing," *Science*, vol. 220, no. 4598, pp. 671–680, 1983.

[57] L. Bottou, "Stochastic gradient descent tricks," in *Neural Networks: Tricks of the Trade - Second Edition*, 2012, pp. 421–436.

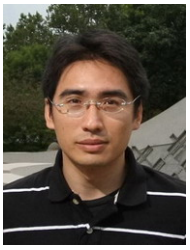
1057-7149 (c) 2018 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

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/TIP.2018.2803341, IEEE Transactions on Image Processing



Daniel Stanley Tan received the M.S. degree in computer science from De La Salle University and is currently a Ph.D. student under the Department of Computer Science and Information Engineering at National Taiwan University of Science and Technology. His research interests include digital image and video processing and computer vision.

Wei-Yang Chen received the B.S. degree in computer science and information engineering from National Taipei University of Technology, Taiwan in 2015. Currently, he is a M.S. student in Department of Computer Science and Information Engineering at National Taiwan University of Science and Technology. His research interests include digital image and video processing and computer vision.



Kai-Lung Hua received the B.S. degree in electrical engineering from National Tsing Hua University in 2000, and the M.S. degree in communication engineering from National Chiao Tung University in 2002, both in Hsinchu, Taiwan. He received the Ph.D. degree from the School of Electrical and Computer Engineering, Purdue University, West Lafayette, IN, in 2010. Since 2010, Dr. Hua has been with National Taiwan University of Science and Technology, where he is currently an associate professor in the Department of Computer Science and Information Engineering. He is a member of Eta Kappa Nu and Phi Tau Phi, as well as a recipient of MediaTek Doctoral Fellowship. His current research interests include digital image and video processing, computer vision, and multimedia networking. He has received several research awards, including Young Scholar Award, Top Performance Award of 2017 ACM Multimedia Grand Challenge, Top 10% Paper Award of 2015 IEEE International Workshop on Multimedia Signal Processing, Second Award of 2014 ACM Multimedia Grand Challenge, Best Paper Award of 2013 IEEE International Symposium on Consumer Electronics, and Best Poster Paper Award of 2012 International Conference on 3D Systems and Applications.

1057-7149 (c) 2018 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.