Research Article

Image Demosaicing Based on Generative Adversarial Network

Jingrui Luo and Jie Wang

School of Automation and Information Engineering, Xi'an University of Technology, Xi'an 710048, China

Correspondence should be addressed to Jie Wang; wangjie7373@163.com

Received 9 April 2020; Accepted 1 June 2020; Published 16 June 2020

Academic Editor: Javier Martinez Torres

Copyright © 2020 Jingrui Luo and Jie Wang. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Digital cameras with a single sensor use a color filter array (CFA) that captures only one color component in each pixel. Therefore, noise and artifacts will be generated when reconstructing the color image, which reduces the resolution of the image. In this paper, we proposed an image demosaicing method based on generative adversarial network (GAN) to obtain high-quality color images. The proposed network does not need any initial interpolation process in the data preparation phase, which can greatly reduce the computational complexity. The generator of the GAN is designed using the U-net to directly generate the demosaicing images. The dense residual network is used for the discriminator to improve the discriminant ability of the network. We compared the proposed method with several interpolation-based algorithms and the DnCNN. Results from the comparative experiments proved that the proposed method can more effectively eliminate the image artifacts and can better recover the color image.

1. Introduction

Images are widely used in people's daily life. Compared to analog images, digital images are more superior in their higher resolution and easier storage, and they are more suitable for computer processing. With the development of computer technology, the digital imaging technology has attracted lots of attention and digital cameras have gradually become the mainstream imaging equipment that is widely used in intelligent transportation [1, 2], medical imaging [3, 4], remote sensing technology [5, 6] and other fields. In our daily life, the digital color images are most commonly used, which include three color components, that is, red, green, and blue, in each pixel. Ideally, digital cameras with three sensors can get full-color images with each sensor capturing one color component, and the three components are combined together into a color image. However, in practice, the arrangement of the three color sensors will affect the subsequent color synthesis, and cameras with three sensors are usually expensive and relatively large. Therefore, most digital cameras use single sensor with a color filter array (CFA) placed in front of the sensor. The obtained CFA image needs to be processed to acquire the full-color image and this process is known as image demosaicing [7]. As only

one color component is captured for each pixel in the CFA, without image demosaicing, the CFA image can only reflect the general outline of the scenery instead of the complete color information, which consequently affects subsequent image processing [8].

The CFA image demosaicing is essentially an ill-posed inverse problem [9]. The methods for image demosaicing generally include interpolation-based algorithms and learning-based algorithms. Generally, image demosaicing using interpolation methods can achieve high accuracy for smooth areas with approximately same colors and gradient brightness. For the color images, the red, green, and blue components occupy different color channels, respectively. When the high-frequency signals change (high-frequency single/information refers to the region with strong color variation, such as edges and angles), there may be spatial offsets in each color channel. Therefore, the reconstructed images may display color artifacts and zippering when doing interpolation [7]. In addition, some traditional interpolation-based methods ignore the correlation among different color channels, which results in unsmooth images [8]. On the whole, the interpolation-based algorithms still have some limitations for image demosaicing, especially at the high-frequency areas.

In recent years, neural networks have been rapidly developed and widely used in image processing, such as image classification [10, 11], motion recognition [12, 13], and image super-resolution [14, 15]. Recently, the generative adversarial network (GAN) [16] has been proposed and rapidly attracts attention of many researchers. Ledig et al. [17] proposed a super-resolution generative adversarial network (SRGAN), which used a deep residual network for the training and can well recover the image textures from greatly downsampled images. Inspired by super-resolution image reconstruction and the conditional generative adversarial network (CGAN), Kupyn et al. [18] applied the CGAN to image deblurring and effectively restored clear images. Pan et al. [19] proposed a physics model constrained learning algorithm so that it can guide the estimation of the specific task in the conventional GAN framework, which can directly solve image restoration problems (such as image deblurring and image denoising).

GAN has been used and played important roles in several areas; however, it has not been used for image demosaicing. In this paper, we proposed a novel learning-based image demosaicing method using GAN to improve the ability for color image recovery. Our contributions are as follows:

- (1) We proposed a CFA image demosaicing method based on GAN
- (2) We carefully designed each part for the GAN model
- (3) We introduced long jump connections for the improved U-net [20] model to design the generator
- (4) We used the dense residual network, which includes dense residual blocks with long jump links and dense connections for the discriminator
- (5) We combined the adversarial loss, the feature loss, and the pixel loss together to further strengthen the network performance

In the experimental section, we show the performance of our method using some comparative experiments. The results prove that the proposed method can more effectively remove artifacts and recover the full-color image, especially for some high-frequency areas such as edges and angles.

2. Related Works

2.1. Interpolation-Based Algorithms. There are many interpolation-based methods for image demosaicing. Linear interpolation algorithm is the simplest one. However, this method often causes artifacts and blurring at the image edges [21]. The bilinear interpolation algorithm [22] estimates the unknown pixels from their adjacent pixels. This method often causes color distortion in the reconstructed image. Malvar et al. [23] proposed a high-quality linear (HQL) interpolation algorithm, which can greatly reduce the computational complexity. However, the artifacts still occur at high-frequency components of the image. In order to further reduce the artifacts, different interpolation techniques were proposed.

Within the gradient-based schemes, Hamilton and Adam [24] proposed the Hamilton–Adam algorithm, which uses the second derivative of the sampled color channels when doing interpolation. Therefore, this method considers the correlation among different color channels and significantly improves the image details. Mukherjee et al. [25] proposed a two-line (TL) interpolation algorithm, which used the homogeneity of the cross-ratios of different spectral components around a small neighborhood to interpolate the pixels lying in the low gradient directions, so as to produce high-quality images.

Within the directional interpolation schemes, Chung and Chan [26] used the prior decision in the horizontal interpolation and the vertical interpolation and got the interpolation result according to the trend of the image edges. This method is prone to producing false colors at tiny edges, especially when the edges are not in the horizontal or vertical directions. Zhang et al. [27] proposed a local directional interpolation and nonlocal adaptive thresholding (LDI-NAT) algorithm. This method used the nonlocal redundancy of the image to improve the local color reproduction and can better reconstruct the edges and reduce color artifacts.

Within the residual interpolation schemes, Kiku et al. [28] proposed a minimized-Laplacian residual interpolation (MLRI) algorithm. This method estimated the tentative pixel values by minimizing the Laplacian energy of the residuals, which can effectively reduce the color artifacts. Monno et al. [29] proposed an adaptive residual interpolation (ARI) algorithm, which adaptively selects a suitable iteration number and combines two different types of residual interpolation algorithms at each pixel. Kiku et al. [30] incorporated the residual interpolation algorithm into the gradient-based threshold free (RI-GBTF) algorithm, and the interpolation accuracy is greatly improved. Besides, L. Zhang and D. Zhang [21] proposed a joint demosaicing-zooming scheme. This method used the correlation of the hyperspectral spatial for the CFA image to calculate the color difference, so as to restore the three color components, which can effectively eliminate color artifacts.

2.2. Learning-Based Algorithms. Recently, neural networks have also been used for image demosaicing. Prakash et al. [31] used a denoising convolution neural network (DnCNN) to perform demosaicing and denoising independently, which effectively suppressed the noise and artifacts. Tan et al. [32] used the deep residual network for image demosaicing and image denoising, which also effectively obtained high-resolution color images. Shopovska et al. [33] proposed an improved residual U-net and used it for image demosaicing, which achieved high-quality reconstructed color images for different CFA patterns. Generally, the learning-based strategies can achieve better performance compared to the traditional interpolation-based methods. However, higher-resolution and clearer recovered color images are the constant pursuit for image demosaicing; that is why we are trying GAN for this task.

3. Problem Formulation

3.1. CFA Image. To obtain a color image with detailed description of the natural image, the best solution is to use three sensors to accept the red, green, and blue components for each pixel, respectively. Then the color image can be synthesized by combining the three color components. Considering the cost and volume, most digital cameras use a single image sensor for the image acquisition systems. The image acquisition of the camera with single sensor is shown in Figure 1. The CFA is set before the sensor. For common CFA, such as the Bayer pattern [34] that is used in this work, the light reaching the sensor mainly consists of the red, green, and blue components. Within the CFA, each pixel only accepts one color component. As shown in Figure 1, the obtained Bayer pattern image can only estimate the approximate gray outline of the scenery instead of the complete color information. The color arrangement of Bayer pattern can be clearly seen from the local zoomed in area. In the Bayer pattern, a set of red and green filters and a set of green and blue filters are alternately used. The number of green pixels is 1/2 of the total number of pixels, while the numbers of red and blue pixels are both 1/4 of the total number of pixels. As only one color component is captured for each pixel, the other two color components need to be recovered according to the color information from adjacent pixels; then a full-color image is obtained from the CFA image. This processing is called image demosaicing.

3.2. Theory of GAN. GAN is a kind of probabilistic generative network, which was first introduced by Goodfellow et al. [16] into the deep learning field. The general architecture of GAN is shown in Figure 2. GAN uses G to perform inverse transformation sampling of the probability distribution and capture the distribution of the ground truth data $\mathbf{x}$. Based on noise data $\mathbf{z}$ which obeys a certain distribution (such as Gaussian distribution), G will generate a fake sample $G(\mathbf{z})$ similar to $\mathbf{x}$. The output of D represents the probability of the incoming data. Thus, if the input is $\mathbf{x}$, the output $D(\mathbf{x})$ is a large probability value; otherwise it outputs a small probability. The GAN's training process is to maximize the discrimination accuracy by training D , as well as to minimize the difference between the generated sample and the real sample by training G . Thus, the training for G and D is a min-max game problem. The performances of G and D are improved by alternative optimization. Finally, G and D reach Nash equilibrium, so that the data distribution synthesized by G is similar to that of the ground truth data $\mathbf{x}$. The loss function of the above process is defined as

$$\min_G \max_D V(D, G) = E_{\mathbf{x} \sim P_{\text{data}}(\mathbf{x})} [\log D(\mathbf{x})] + E_{\mathbf{z} \sim P_{\text{noise}}(\mathbf{z})} [\log (1 - D(G(\mathbf{z})))] \quad (1)$$

where $V(D, G)$ represents the value function [16]. $\mathbf{x} \sim P_{\text{data}}(\mathbf{x})$ represents ground truth data $\mathbf{x}$ obeying a real data distribution $P_{\text{data}}(\mathbf{x})$, and $\mathbf{z} \sim P_{\text{noise}}(\mathbf{z})$ represents noise data $\mathbf{z}$ obeying a simulated distribution $P_{\text{noise}}(\mathbf{z})$ (such as the Gaussian distribution). $D(\mathbf{x})$ and $D(G(\mathbf{z}))$ are the

classification outputs of D for the ground truth data $\mathbf{x}$ and the generated data $G(\mathbf{z})$, respectively. E means expectation.

4. The Proposed Method

In this section, we propose an effective demosaicing algorithm based on GAN. The whole process is shown in Figure 3. The proposed algorithm first extracts the red, green, and blue components from the original CFA image to form the 3-channeled split CFA image. Then the extracted green component is further separated into two channels to form the 4-channeled split CFA image. Subsequently, the algorithm extracts only the pixel values that are not 0 to compress the 4-channeled split CFA image. The compressed 4-channeled image is taken as the input of G in GAN. The output of G is the interpolated 3-channel full-color image. The output images from G and the ground truth images are then inserted into D . The parameters of G are optimized according to the output of D . We designed the architectures for G and D and trained the database through an end-to-end trainable neural network. In addition, the algorithm combined the adversarial loss, pixel loss, and feature loss to design the generator loss function in order to further improve the network performance [35]. In the following, we give a detailed introduction to different parts of the network.

4.1. Generator. The purpose of the generator G is to convert the 4-channeled compressed CFA images to the 3-channeled output full-color images. The structure of G is shown in Figure 4. We used the improved U-net [20] model for G . Overall, the generator G consists of an encoder (the first half) and a decoder (the second half), which is shown in Figure 4. One layer in the encoder and the corresponding layer in the decoder form a U-shaped symmetric layer. The long jump links within each symmetric layer in the U-net model can reduce the information redundancy. Besides, we remove the pooling layer in the U-net, which can avoid the loss of useful information in the feature maps and increase the stability of the training process.

The encoder is mainly based on the downsampling operation (i.e., convolution operation). It can analyze the input data to obtain the most significant features and provide feature mappings to its corresponding layer in the decoder. The activation function of the encoder is a leaky rectified linear unit (LReLU), which is defined as

$$\text{LReLU}(\mathbf{x}_e) = \max(0, \mathbf{x}_e) + \kappa \times \min(0, \mathbf{x}_e), \quad (2)$$

where κ is a positive constant ($\kappa \in (0, 1)$). $\mathbf{x}_e$ represents the input vectors for a specific layer of the encoder. In our experiments, we set κ as 0.1.

The decoder is mainly based on the upsampling operation (i.e., deconvolution operation) to restore the full-color images. The activation function of the decoder is a standard rectifier linear unit (ReLU), which is defined as

$$\text{ReLU}(\mathbf{x}_d) = \max(0, \mathbf{x}_d), \quad (3)$$

where $\mathbf{x}_d$ represents the input vectors for a specific layer of the decoder.

FIGURE 1: Image acquisition of the digital cameras with single sensor.

FIGURE 2: The general architecture of GAN.

FIGURE 3: The process of the proposed CFA image demosaicing.

FIGURE 4: The structure of the generator.

Particularly for the final layer of the decoder, the activation function is the tanh activation function, which is defined as

$$\tanh(\mathbf{x}_f) = \frac{e^{\mathbf{x}_f} - e^{-\mathbf{x}_f}}{e^{\mathbf{x}_f} + e^{-\mathbf{x}_f}}, \quad (4)$$

where $\mathbf{x}_f$ represents the input vectors for the final layer of the decoder.

In order to accelerate the convergence and improve the network performance, we introduce the batch normalization (BN) operation after each convolution and deconvolution operation to slow down the transfer of internal covariates and reduce the sensitivity of the network to initialization weights [36].

Detailed parameters for the convolution and deconvolution layers are shown in Table 1.

TABLE 1: The parameters for the generator.

Encoder	Kernel	3 × 3	4 × 4	4 × 4	Convolution				
	Stride	1 × 1	2 × 2	2 × 2	4 × 4	4 × 4	4 × 4	4 × 4	4 × 4
	Channel	16	32	64	2 × 2	2 × 2	2 × 2	2 × 2	2 × 2
Decoder	Kernel	4 × 4	4 × 4	4 × 4	Deconvolution				
	Stride	2 × 2	2 × 2	2 × 2	4 × 4	4 × 4	4 × 4	4 × 4	3 × 3
	Channel	512	512	256	2 × 2	2 × 2	2 × 2	2 × 2	1 × 1
					256	128	64	32	3

4.2. Discriminator. We used a dense residual network, which is inspired by the ResNet [36], for the discriminator D . The ResNet is formed by stacking multiple consecutive residual blocks (RB). In order to improve the network performance and solve the problem of gradient disappearance and gradient dispersion during the network training, we used an improved residual dense block (RDB). The structure of D is shown in Figure 5. The long jump connection after each RDB helps to transfer the output of this RDB to the final convolution layer. Within each RDB, there are several units with each unit consisting of the ReLU activation function, the convolution layer, and the BN operation. There are dense connections with different distances among these units. The output from the final convolution layer is mapped into 0 or 1 using the sigmoid activate function. The sigmoid function performs a probability analysis that can normalize the discriminant result, which is defined as

$$\text{sigmoid}(\mathbf{x}_s) = \frac{1}{(1 + e^{-\mathbf{x}_s})}, \quad (5)$$

where $\mathbf{x}_s$ represents the input vectors for the sigmoid function.

For the convolution layers in D , the kernel size is set as 3×3 , the stride size is set as 1×1 , and an output channel is 64.

4.3. Loss Function. Denote the ground truth images $\{\mathbf{x}_i, i = 1, 2, \dots, N\}$, where N represents the number of images. After a series of operations, the CFA images are transformed into the corresponding 4-channeled compressed CFA images, which are denoted as $\{\mathbf{y}_i, i = 1, 2, \dots, N\}$ and are regarded as the input of G . According to the loss function inspired by Alsaiani et al. [35], we combined the adversarial loss, the feature loss, and the pixel loss together with appropriate weights to work as the final loss function for the generator. The adversarial loss function (L_a) is expressed as

$$L_a = \frac{1}{N} \left(\sum_{i=1}^N \log[1 - D(G(\mathbf{y}_i))] \right), \quad (6)$$

where $\mathbf{y}_i$ represents the i -th 4-channeled compressed CFA image. The 3-channeled color images are produced using Equation (6) to fool the generator G .

The feature loss function (L_f) is defined as

$$L_f = \frac{1}{N} \sum_{i=1}^N \left(\|F[G(\mathbf{y}_i)] - F(\mathbf{x}_i)\|^2 \right), \quad (7)$$

where $F(\cdot)$ represents the feature mapping matrix extracted from the pretrained VGG network [35]. $\|\cdot\|^2$ represents the L2 norm. Using Equation (7), we can extract the image features and restore the image details by comparing feature data between the generated image $G(\mathbf{y}_i)$ and the ground truth image $\mathbf{x}_i$.

The pixel loss (pixel-to-pixel Euclidean distance) function (L_p) is defined as

$$L_p = \frac{1}{N} \sum_{i=1}^N \left(\|G(\mathbf{y}_i) - \mathbf{x}_i\|^2 \right) + \lambda \|\nabla G(\mathbf{y}_i)\|^2, \quad (8)$$

where $\|\nabla G(\mathbf{y}_i)\|^2$ is the regularization item, with λ representing the regularization weight. Using Equation (8), we can correctly restore the image information by comparing each pixel between the generated image $G(\mathbf{y}_i)$ and the ground truth image $\mathbf{x}_i$.

We combine L_a , L_f , and L_p together with appropriate weights to form the final loss function for the generator, which is defined as

$$L_G = \alpha L_a + \beta L_f + \gamma L_p, \quad (9)$$

where α , β , and γ represent the predefined positive weights according to the empirical values [35].

According to Equation (1), the discriminator D uses the following equation to update the parameters:

$$L_D = \max_D \frac{1}{N} \sum_{i=1}^N \{ \log[D(\mathbf{x}_i)] + \log[1 - D(G(\mathbf{y}_i))] \}. \quad (10)$$

For the ground truth image $\mathbf{x}_i$, the probability of the output $D(\mathbf{x}_i)$ is close to 1. For the generated image $G(\mathbf{y}_i)$, the probability of the output $D(G(\mathbf{y}_i))$ is close to 0.

Based on the above strategy, the generator G and the discriminator D will be alternately optimized.

Based on the above introduction, we give the whole pipeline in Figure 6 to clearly describe the proposed method. The real scenery is captured by the camera and converted to the CFA image. The obtained CFA image is then further converted to the 4-channeled compressed CFA images, which is inserted into the generator designed by the U-net model. The output from the generator and the ground truth image are then inputted into the discriminator that is designed by the dense residual network. The generator will finally give near-real demosaiced image through the network training.

5. Experiments

In this section, we demonstrate the performance of the proposed network with numerical experiments. The network

FIGURE 5: The structure of the discriminator.

FIGURE 6: The pipeline of the proposed method.

FIGURE 7: The operation of data augmentation.

training is carried out under the TensorFlow environment, which is installed on a PC with Nvidia GeForce® MX250 GPU and Intel Core i5-8265U CPU. The training sets are created beforehand and then uploaded into TensorFlow.

5.1. Training Details. The training database used in this paper is from the Waterloo Exploration database (WED) [37], which contains 4744 pristine natural images. We first

randomly selected 400 images to create the training set. For the training set, we used data augmentation operations such as cropping and rotations to increase the number of images. To be more specific, we first scaled down each selected image by 1, 0.9, 0.8, and 0.7 times and then used a sliding window to crop the scaled images into patches with a size of 40×40 pixels. The sliding step-lengths in the horizontal and vertical directions are both 20 pixels. Subsequently, the obtained

TABLE 2: CPSNR and SSIM of the test results for the Kodak database by different methods.

No.	1		2		3		4		5		6		7		8	
	CPSNR	SSIM	CPSNR	SSIM	CPSNR	SSIM	CPSNR	SSIM	CPSNR	SSIM	CPSNR	SSIM	CPSNR	SSIM	CPSNR	SSIM
Bilinear	33.55	0.8060	38.66	0.8984	39.59	0.9333	38.56	0.9118	34.22	0.8777	34.94	0.8443	39.16	0.9538	33.19	0.8243
TL	32.98	0.8081	36.91	0.8564	38.64	0.9201	37.16	0.8788	33.76	0.8500	34.14	0.8364	37.06	0.9288	32.75	0.8323
HQL	36.02	0.9505	41.39	0.9602	43.13	0.9761	42.32	0.9709	37.30	0.9701	37.40	0.9566	42.47	0.9827	35.17	0.9502
Zhang's	38.79	0.9881	40.46	0.9754	41.45	0.9861	40.11	0.9793	37.05	0.9859	40.23	0.9886	40.25	0.9877	35.99	0.9865
LDI-NAT	33.69	0.9633	39.01	0.9683	40.40	0.9802	39.02	0.9718	34.77	0.9797	34.90	0.9693	40.93	0.9869	32.01	0.9676
ARI	38.84	0.9863	39.69	0.9704	42.75	0.9874	40.66	0.9904	38.29	0.9887	40.55	0.9885	42.74	0.9888	34.99	0.9801
MLRI	36.86	0.9818	41.49	0.9807	42.95	0.9883	41.99	0.9842	37.68	0.9885	38.99	0.9865	42.79	0.9904	35.17	0.9825
RI-GBTF	39.03	0.9880	40.08	0.9753	41.51	0.9870	40.49	0.9803	37.81	0.9878	40.89	0.9899	40.93	0.9869	32.01	0.9676
DnCNN	41.96	0.9874	43.73	0.9732	42.18	0.9869	43.26	0.9867	37.77	0.9849	40.26	0.9873	42.08	0.9873	36.49	0.9738
Proposed	42.07	0.9882	43.83	0.9757	42.22	0.9876	43.93	0.9880	39.49	0.9897	41.77	0.9901	43.53	0.9894	36.88	0.9869
No.	9		10		11		12		13		14		15		16	
	CPSNR	SSIM	CPSNR	SSIM	CPSNR	SSIM	CPSNR	SSIM	CPSNR	SSIM	CPSNR	SSIM	CPSNR	SSIM	CPSNR	SSIM
Bilinear	38.58	0.9198	38.82	0.9191	36.02	0.8680	38.89	0.9063	32.51	0.7690	35.54	0.8709	38.76	0.9174	36.88	0.8748
TL	38.18	0.9156	38.07	0.9083	35.34	0.8545	37.80	0.8975	32.32	0.7867	34.58	0.8467	37.97	0.9058	35.80	0.8601
HQL	41.59	0.9702	42.41	0.9722	38.75	0.9611	42.41	0.9697	35.03	0.9420	38.54	0.9616	41.74	0.9659	39.87	0.9629
Zhang's	42.27	0.9848	42.04	0.9858	39.85	0.9868	42.83	0.9868	35.01	0.9821	35.52	0.9806	39.12	0.9758	43.73	0.9900
LDI-NAT	40.11	0.9744	39.81	0.9778	36.13	0.9728	40.72	0.9777	29.64	0.9461	35.41	0.9718	37.58	0.9671	38.56	0.9746
ARI	41.96	0.9714	41.67	0.9819	39.67	0.9843	43.25	0.9869	35.22	0.9804	37.57	0.9836	38.64	0.9715	43.33	0.9892
MLRI	42.36	0.9817	42.44	0.9847	39.36	0.9863	43.37	0.9868	33.15	0.9732	37.75	0.9834	39.85	0.9777	42.84	0.9885
RI-GBTF	42.54	0.9848	42.61	0.9866	40.24	0.9880	43.39	0.9876	35.32	0.9824	36.31	0.9817	38.90	0.9738	44.38	0.9901
DnCNN	42.76	0.9857	40.04	0.9836	42.32	0.9875	42.15	0.9870	36.94	0.9837	39.50	0.9803	40.95	0.9725	44.65	0.9913
Proposed	43.43	0.9883	43.60	0.9871	42.66	0.9901	43.99	0.9894	37.03	0.9861	40.02	0.9845	42.36	0.9779	44.96	0.9904
No.	17		18		19		20		21		22		23		24	
	CPSNR	SSIM	CPSNR	SSIM	CPSNR	SSIM	CPSNR	SSIM	CPSNR	SSIM	CPSNR	SSIM	CPSNR	SSIM	CPSNR	SSIM
Bilinear	38.58	0.9244	35.55	0.8730	36.01	0.8750	38.53	0.9198	35.81	0.8878	36.65	0.8816	41.32	0.9551	35.53	0.8774
TL	37.82	0.9136	34.84	0.8595	35.46	0.8798	38.04	0.9132	35.24	0.8720	36.01	0.8779	40.03	0.9399	35.24	0.8790
HQL	42.17	0.9760	38.50	0.9612	38.20	0.9619	41.27	0.9651	38.42	0.9640	37.78	0.9606	44.46	0.9793	38.54	0.9663
Zhang's	41.67	0.9882	37.13	0.9794	40.68	0.9856	40.94	0.9769	39.12	0.9824	37.96	0.9741	41.99	0.9820	35.38	0.9853
LDI-NAT	38.31	0.9789	33.89	0.9653	37.22	0.9692	38.31	0.9672	35.22	0.9686	37.11	0.9719	41.80	0.9811	31.77	0.9731
ARI	41.32	0.9868	36.91	0.9753	40.65	0.9821	40.89	0.9760	39.28	0.9748	38.08	0.9728	43.24	0.9814	35.36	0.9849
MLRI	40.69	0.9865	36.55	0.9782	40.03	0.9823	40.71	0.9770	38.17	0.9802	38.55	0.9772	43.67	0.9857	34.64	0.9842
RI-GBTF	41.75	0.9890	37.38	0.9805	41.12	0.9860	41.21	0.9780	39.63	0.9833	38.36	0.9763	41.96	0.9816	35.01	0.9858
DnCNN	42.18	0.9849	38.96	0.9815	42.74	0.9847	40.55	0.9873	40.08	0.9825	39.79	0.9805	44.67	0.9843	39.13	0.9867
Proposed	42.25	0.9891	39.47	0.9983	42.96	0.9926	42.86	0.9877	40.87	0.9894	40.37	0.9862	44.98	0.9916	40.07	0.9954

FIGURE 8: Box plots of (a) CPSNR and (b) SSIM for the Kodak database by different methods. The middle red line in each box represents the median value. The red plus symbol represents the outlier.

patches are sequentially vertically and horizontally flipped and rotated 90° , 180° , and 270° , respectively, as shown in Figure 7. Through the above data augmentation operations, we obtained 86400 training images. These images are input in batches during the network training process to reduce the calculation and avoid local extreme value problems.

During the training, the weighting parameters in the loss functions are set as $\alpha = 0.5$, $\beta = 1$, $\gamma = 1$, and $\lambda = 10^{-5}$ according to the empirical values [35]. The batch size is set as 256 and there are 200 iterations for the whole network training. During the training, we used a variable learning rate, where the initial learning rate is set as 0.01 and the value is reduced by 1/10 every 40 iterations. The trained network is tested with the Kodak database and the McMaster database [38]. The Kodak database consists of 24 images with the size of 768×512 pixels. The McMaster database consists of 18 images with the size of 500×500 pixels.

In order to quantitatively evaluate the performance of the proposed network, we used color peak signal to noise ratio (CPSNR) and structural similarity index (SSIM) as measurement standards for the demosaicing results. The CPSNR value is calculated as

$$\text{CPSNR} = 10 \log \frac{(255)^2}{(1/3 * H * W) \sum_i^3 \sum_{h=0}^{H-1} \sum_{w=0}^{W-1} (y(h, w, t) - x(h, w, t))^2}, \quad (11)$$

where $x(h, w, t)$ and $y(h, w, t)$ represent the (h, w) pixel value of the ground truth image $\mathbf{x}$ and the demosaiced image $\mathbf{y}$ for the t -th color channel, respectively. H and W represent the height size and width size of the image.

The SSIM measures the similarity between two images, which is defined as

$$\text{SSIM}(\mathbf{x}, \mathbf{y}) = \frac{(2\mu_x\mu_y + C_1)(\sigma_{xy} + C_2)}{(\mu_x^2 + \mu_y^2 + C_1)(\sigma_x\sigma_y + C_2)}, \quad (12)$$

where μ_x and σ_x represent the mean intensity and the standard deviation of the ground truth image $\mathbf{x}$. μ_y and σ_y represent the mean intensity and the standard deviation of the demosaiced image $\mathbf{y}$; σ_{xy} is the covariance between $\mathbf{x}$ and $\mathbf{y}$. C_1 and C_2 are two constants used to keep the equation balanced and stable, which are usually set as $C_1 = (K_1 * K)^2$ and $C_2 = (K_2 * K)^2$, with $K_1 = 0.01$, $K_2 = 0.03$, and $K = 255$.

5.2. Image Demosaicing Test. In this section, we prove the effectiveness of the proposed method by comparing different demosaicing methods. The methods used for comparison are the Bilinear [22], TL [25], HQL [23], Zhang's [21], LDI-NAT [27], ARI [29], MLRI [28], RI-GBTF [30], and DnCNN [31] methods, as well as the proposed method.

Table 2 shows the CPSNR and SSIM of the test results for the Kodak database from different methods. Figure 8 shows the corresponding box plots of the CPSNR and SSIM for easier comparison. Table 3 shows the CPSNR and SSIM of the test results for the McMaster database from different methods. Similarly, Figure 9 shows the corresponding box plots. We can see that the proposed method shows higher CPSNR and lower SSIM, which means better performance compared with other methods.

For additional comparison, Figure 10 shows the reconstructed images using different methods for the 19th image in the Kodak database. The marked portion of the image within the black box (the fence) is enlarged for clearer comparison. It can be seen that this part has obvious vertical textures and it is prone to artifacts. Residual images (i.e., difference between the ground truth image and the demosaiced images) for the enlarged portion are also shown for easier comparison. From the reconstructed images and the residual images, we can see that, compared with other methods, the proposed method can more effectively

TABLE 3: CPSNR and SSIM of the test results for the McMaster database by different methods.

No.	1		2		3		4		5		6		7		8		9	
	CPSNR	SSIM	CPSNR	SSIM	CPSNR	SSIM	CPSNR	SSIM	CPSNR	SSIM	CPSNR	SSIM	CPSNR	SSIM	CPSNR	SSIM	CPSNR	SSIM
Bilinear	29.58	0.8939	33.39	0.9252	28.99	0.9169	31.55	0.9564	34.65	0.9374	37.83	0.9599	31.78	0.8882	33.10	0.9486	34.76	0.9498
TL	33.67	0.8261	35.72	0.8740	33.96	0.8753	35.96	0.9422	35.91	0.8715	36.29	0.8887	34.81	0.8662	37.39	0.9254	37.03	0.9047
HQL	34.10	0.8733	37.33	0.9172	35.72	0.9501	37.93	0.9784	37.66	0.9239	38.23	0.9389	38.22	0.9537	39.73	0.9607	38.76	0.9373
Zhang's	25.97	0.8484	32.89	0.9135	31.99	0.9503	33.76	0.9811	29.89	0.8999	31.85	0.9000	38.71	0.9734	36.73	0.9610	33.14	0.9237
LDI-NAT	27.47	0.8760	33.98	0.9271	31.66	0.9542	34.86	0.9842	32.58	0.9285	35.61	0.9420	35.14	0.9546	36.65	0.9656	35.51	0.9449
ARI	29.63	0.9243	35.23	0.9455	34.65	0.9728	37.92	0.9894	35.44	0.9625	38.79	0.9626	39.75	0.9792	39.55	0.9784	37.90	0.9630
MLRI	28.98	0.9135	35.06	0.9417	33.85	0.9697	37.66	0.9893	33.94	0.9533	38.29	0.9660	37.50	0.9705	37.05	0.9710	36.49	0.9588
RI-GBTF	27.44	0.8869	33.64	0.9235	32.71	0.9603	34.85	0.9843	31.43	0.9322	34.47	0.9431	39.11	0.9754	36.36	0.9640	34.10	0.9429
DnCNN	29.39	0.9236	36.66	0.9358	35.86	0.9673	37.74	0.9921	36.28	0.9579	37.95	0.9642	39.30	0.9735	38.45	0.9691	38.51	0.9582
Proposed	34.29	0.9367	37.67	0.9486	35.88	0.9738	38.06	0.9980	37.70	0.9635	38.93	0.9693	39.73	0.9798	39.87	0.9804	39.74	0.9679

No.	10		11		12		13		14		15		16		17		18	
	CPSNR	SSIM	CPSNR	SSIM	CPSNR	SSIM	CPSNR	SSIM	CPSNR	SSIM	CPSNR	SSIM	CPSNR	SSIM	CPSNR	SSIM	CPSNR	SSIM
Bilinear	36.86	0.9543	36.95	0.9495	34.62	0.9520	39.04	0.9606	37.55	0.9487	37.66	0.9447	31.45	0.9278	34.75	0.9375	31.46	0.9098
TL	37.10	0.8960	38.44	0.9206	37.40	0.9021	39.64	0.9153	38.79	0.9147	39.10	0.9172	34.79	0.8799	34.32	0.8727	35.56	0.8756
HQL	40.17	0.9539	40.51	0.9451	40.06	0.9517	40.04	0.9434	40.79	0.9426	40.77	0.9360	36.33	0.9051	36.12	0.8963	37.32	0.9361
Zhang's	35.01	0.9402	35.91	0.9273	35.58	0.9446	37.73	0.9381	36.49	0.9368	36.36	0.9257	29.03	0.8675	27.65	0.8324	33.24	0.9379
LDI-NAT	37.24	0.9560	37.57	0.9427	37.33	0.9547	39.72	0.9488	37.76	0.9471	37.83	0.9414	31.01	0.8913	30.81	0.9031	33.71	0.9397
ARI	39.28	0.9723	40.23	0.9730	40.07	0.9647	40.64	0.9507	39.05	0.9575	39.41	0.9593	35.64	0.9673	34.71	0.9616	36.43	0.9666
MLRI	38.65	0.9688	39.98	0.9714	39.65	0.9636	40.63	0.9531	38.81	0.9552	38.92	0.9528	35.16	0.9632	32.58	0.9415	36.12	0.9649
RI-GBTF	36.27	0.9552	37.85	0.9610	38.13	0.9579	38.76	0.9448	37.27	0.9431	36.65	0.9309	32.19	0.9442	29.06	0.9392	35.28	0.9590
DnCNN	39.04	0.9678	39.47	0.9746	40.03	0.9614	39.87	0.9525	39.36	0.9579	40.18	0.9487	35.94	0.9603	33.89	0.9589	37.21	0.9631
Proposed	40.35	0.9755	40.62	0.9787	40.11	0.9675	40.71	0.9616	40.95	0.9670	40.93	0.9603	36.82	0.9679	34.64	0.9677	37.65	0.9690

FIGURE 9: Box plots of (a) CPSNR and (b) SSIM for the McMaster database by different methods. The middle red line in each box represents the median value. The red plus symbol represents the outlier.

FIGURE 10: Comparison of reconstructed images from different demosaicing methods for the 19th image in the Kodak database. (a) Ground truth image; (b) enlarged portion in the green box; (c–l) the results by using Bilinear, TL, HQL, Zhang's, LDI-NAT, ARI, MLRI, RI-GBTF, and DnCNN methods, as well as the proposed method, respectively; (m–v) the residual images corresponding to (c–l).

suppress the artifact phenomenon, especially for some tiny edges and angle areas.

Figure 11 shows the reconstructed images using different methods for the 22th image in the Kodak database. The marked portion in the black box (the window) is enlarged. It can be seen that this part is prone to appearing color stripes and zipperings. The residual images are also shown for easier comparison. From the reconstructed images and the residual images, we can see that most methods can obtain satisfactory results for the smooth areas, while there may appear some wrong colors at the edges. From the comparison, we can see

that the results of the proposed method show relatively fewer artifacts and color stripes.

Figures 12 and 13 show the reconstructed images using different methods for the 1st and 12th images in the McMaster database, respectively. Similarly, the marked portions in the images are enlarged and the residual images for the enlarged portion are shown for clear comparisons. It can be seen that, compared with other methods, the proposed method can better recover the images with fewer artifacts, especially at some tiny edges, which proved the validity and performance of the proposed method.

FIGURE 11: Comparison of reconstructed images from different demosaicing methods for the 22th image in the Kodak database. (a) Ground truth image; (b) enlarged portion in the green box; (c–l) the results by using Bilinear, TL, HQL, Zhang's, LDI-NAT, ARI, MLRI, RI-GBTF, and DnCNN methods, as well as the proposed method, respectively; (m–v) the residual images corresponding to (c–l).

FIGURE 12: Comparison of reconstructed images from different demosaicing methods for the 1st image in the McMaster database. (a) Ground truth image; (b) enlarged portion in the black box; (c–l) the results by using Bilinear, TL, HQL, Zhang's, LDI-NAT, ARI, MLRI, RI-GBTF, and DnCNN methods, as well as the proposed method, respectively; (m–v) the residual images corresponding to (c–l).

6. Discussion

In this work, we proposed a new method for image demosaicing based on GAN, which aims to more effectively reconstruct the full-color image. One of the challenges of this task is the recovery of the high-frequency information in the image, such as edges and angles. Many related algorithms have strong ability to process the smooth part of the image; however artifacts, zippering, and strip colors still exist in the high-frequency part. In the current work, we redesigned the generator and discriminator of GAN and combined the adversarial loss, the feature loss, and the pixel loss to further improve the network performance. Numerical experiments

showed that the proposed algorithm can effectively reduce the artifacts at the edges and produce near-real reconstructed images, which can be the basis for subsequent image processing, such as image recognition and image transmission. The proposed method can produce better recovered color images; however, the learning-based strategy is relatively time-consuming in the data training phase. Therefore, how to improve the efficiency of the network training is an important aspect to further enhance the performance of the learning-based technology. In practice, there are many kinds of CFA patterns. We used the Bayer pattern in this paper. Different patterns of the CFA image may have different impact on the reconstructed image, so we will use CFA

FIGURE 13: Comparison of reconstructed images from different demosaicing methods for the 12th image in the McMaster database. (a) Ground truth image; (b) enlarged portion in the green box; (c–l) the results by using Bilinear, TL, HQL, Zhang's, LDI-NAT, ARI, MLRI, RI-GBTF, and DnCNN methods, as well as the proposed method, respectively; (m–v) the residual images corresponding to (c–l).

images with different designs to test the network in the near future. In the current work, we assumed the CFA images are noiseless. However, the images from cameras in practice may have been affected by noises. Therefore, we will try combining image demosaicing and denoising in the future. The current work focuses on directly generating the demosaiced images using neural network. We will test on combining traditional demosaicing algorithms and the neural network in the future.

7. Conclusions

In this paper, we proposed an image demosaicing method based on GAN. The generator is designed by using the improved U-net architecture to directly generate the demosaicing images. For the discriminator, we used the dense residual network including dense residual blocks with long jump connections and dense connections to overcome the problem of gradient disappearance and gradient dispersion during the network training, which can improve the discriminant ability of the network. In addition, we combined the adversarial loss, the pixel loss, and the feature loss together to improve the loss function. The network was trained using images from the Waterloo Exploration database and the trained network was tested with the Kodak database and the McMaster database. Comparisons among different image demosaicing methods showed that the proposed method can better eliminate artifacts in the reconstructed image and can especially better restore high-frequency features, such as edges and angles of the image.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request. The data used to support the findings of this study are open datasets which could be found in general websites, and the datasets are also freely available.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

The authors acknowledge the National Natural Science Foundation of China (Grant no. 41704118) and the Natural Science Basic Research Plan in Shaanxi Province of China (Grant no. 2020JM-446).

References

- [1] Y. Yuan and F. Y. Wang, "Towards blockchain-based intelligent transportation systems," in *Proceedings of the IEEE International Conference on Intelligent Transportation Systems*, pp. 2663–2668, IEEE, Rio de Janeiro, Brazil, November 2016.
- [2] C. N. E. Anagnostopoulos, I. E. Anagnostopoulos, V. Loumos, and E. Kayafas, "A license plate-recognition algorithm for intelligent transportation system applications," *IEEE Transactions on Intelligent Transportation Systems*, vol. 7, no. 3, pp. 377–392, 2006.
- [3] J. Rosen and G. Brooker, "Digital spatially incoherent Fresnel holography," *Optics Letters*, vol. 32, no. 8, pp. 912–914, 2007.
- [4] E. R. Hunt, M. Cavigelli, C. S. T. Daughtry, J. E. Mcmurtrey, and C. L. Walthall, "Evaluation of digital photography from model aircraft for remote sensing of crop biomass and nitrogen status," *Precision Agriculture*, vol. 6, no. 4, pp. 359–378, 2005.
- [5] A. D. Richardson, B. H. Braswell, D. Y. Hollinger, J. P. Jenkins, and S. V. Ollinger, "Near-surface remote sensing of spatial and temporal variation in canopy phenology," *Ecological Applications*, vol. 19, no. 6, pp. 1417–1428, 2009.
- [6] A. Rango, A. Laliberte, J. E. Herrick, C. Winters, and D. Browning, "Unmanned aerial vehicle-based remote sensing for rangeland assessment, monitoring, and management," *Journal of Applied Remote Sensing*, vol. 3, no. 1, Article ID 033542, 2009.
- [7] T. Yamaguchi and M. Ikehara, "Image demosaicking via chrominance images with parallel convolutional neural

- networks,” in *Proceedings of the 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 1702–1706, Brighton, UK, May 2019.
- [8] D. Khashabi, S. Nowozin, J. Jancsary, and A. W. Fitzgibbon, “Joint demosaicing and denoising via learned nonparametric random fields,” *IEEE Transactions on Image Processing*, vol. 23, no. 12, pp. 4968–4981, 2014.
 - [9] J. Wang, C. Zhang, and P. Hao, “New color filter arrays of high light sensitivity and high demosaicking performance,” in *Proceedings of the 2011 IEEE International Conference on Image Processing*, pp. 3153–3156, Brussels, Belgium, September 2011.
 - [10] Y. Han, T. Jiang, Y. Ma, and C. Xu, “Pretraining convolutional neural networks for image -based vehicle classification,” *Advances in Multimedia*, vol. 2018, Article ID 3138278, 10 pages, 2018.
 - [11] C. Tang, Q. Zhu, W. Wu, W. Huang, C. Hong, and X. Niu, “PLANET: improved convolutional neural networks with image enhancement for image classification,” *Mathematical Problems in Engineering*, vol. 2020, Article ID 1245924, 10 pages, 2020.
 - [12] P. Wang, W. Li, Z. Gao, J. Zhang, C. Tang, and P. O. Ogunbona, “Action recognition from depth maps using deep convolutional neural networks,” *IEEE Transactions on Human-Machine Systems*, vol. 46, no. 4, pp. 498–509, 2016.
 - [13] Y. Hou, Z. Li, P. Wang, and W. Li, “Skeleton optical spectra-based action recognition using convolutional neural networks,” *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 28, no. 3, pp. 807–811, 2018.
 - [14] Z. Hua, H. Zhang, and J. Li, “Image super resolution using fractal coding and residual network,” *Complexity*, vol. 2019, Article ID 9419107, 14 pages, 2019.
 - [15] X. Zhu, L. Zhang, L. Zhang, X. Liu, Y. Shen, and S. Zhao, “GAN-based image super-resolution with a novel quality loss,” *Mathematical Problems in Engineering*, vol. 2020, Article ID 9419107, 12 pages, 2020.
 - [16] I. Goodfellow, J. Pouget-Abadie, M. Mirza et al., “Generative adversarial nets,” in *Proceedings of the Advances in Neural Information Processing Systems*, pp. 2672–2680, Montreal, Canada, 2014.
 - [17] C. Ledig, L. Theis, F. Huszár et al., “Photo-realistic single image super-resolution using a generative adversarial network,” 2017, <https://arxiv.org/abs/1609.04802>.
 - [18] O. Kupyn, V. Budzan, M. Mykhailych, D. Mishkin, and J. Matas, “DeblurGAN: blind motion deblurring using conditional adversarial networks,” 2017, <https://arxiv.org/abs/1711.07064>.
 - [19] J. Pan, Y. Liu, and J. Dong, “Physics-based generative adversarial models for image restoration and beyond,” 2018, <https://arxiv.org/abs/1808.00605>.
 - [20] O. Ronneberger, P. Fischer, and T. Brox, “U-net: convolutional networks for biomedical image segmentation,” in *Proceedings of the International Conference on Medical Image Computing and Computer-Assisted Intervention*, pp. 234–241, Munich, Germany, October 2015.
 - [21] L. Zhang and D. Zhang, “A joint demosaicking–zooming scheme for single chip digital color cameras,” *Computer Vision and Image Understanding*, vol. 107, no. 1–2, pp. 14–25, 2007.
 - [22] 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, 2003.
 - [23] H. S. Malvar, L. He, and R. Cutler, “High-quality linear interpolation for demosaicing of Bayer-patterned color images,” in *Proceedings of the 2004 IEEE International Conference on Acoustics, Speech, and Signal Processing*, pp. 485–488, Montreal, Canada, May 2004.
 - [24] J. F. Hamilton and J. E. Adam, “Adaptive color plane interpolation in single sensor color electronic camera,” US Patent No. 5629734, 1997.
 - [25] J. Mukherjee, M. S. Moore, and S. K. Mitra, “Color demosaicing with constrained buffering,” in *Proceedings of the Sixth International Symposium on Signal Processing and Its Applications (Cat.No.01EX467)*, pp. 52–55, Kuala Lumpur, Malaysia, August 2001.
 - [26] K.-H. Chung and Y.-H. Chan, “Color demosaicing using variance of color differences,” *IEEE Transactions on Image Processing*, vol. 15, no. 10, pp. 2944–2955, 2006.
 - [27] L. Zhang, X. Wu, A. Buades, and X. Li, “Color demosaicking by local directional interpolation and nonlocal adaptive thresholding,” *Journal of Electronic Imaging*, vol. 20, no. 2, Article ID 023016, 2011.
 - [28] D. Kiku, Y. Monno, M. Tanaka, and M. Okutomi, “Minimized-Laplacian residual interpolation for color image demosaicking,” in *Proceedings of the SPIE*, San Francisco, CA, USA, March 2014.
 - [29] Y. Monno, D. Kiku, M. Tanaka, and M. Okutomi, “Adaptive residual interpolation for color image demosaicking,” in *Proceedings of the 2015 IEEE International Conference on Image Processing (ICIP)*, pp. 3861–3865, Quebec, Canada, September 2015.
 - [30] D. Kiku, Y. Monno, M. Tanaka, and M. Okutomi, “Beyond color difference: residual interpolation for color image demosaicking,” *IEEE Transactions on Image Processing: A Publication of the IEEE Signal Processing Society*, vol. 25, no. 3, pp. 1288–1300, 2016.
 - [31] V. Prakash, K. S. Prasad, and T. J. C. Prasad, “Deep learning approach for image denoising and image demosaicing,” *International Journal of Computer Applications*, vol. 168, no. 9, pp. 18–26, 2017.
 - [32] H. Tan, H. Xiao, S. Lai, Y. Liu, and M. Zhang, “Deep residual learning for image demosaicing and blind denoising,” *Pattern Recognition Letters*, vol. 2018, 2018.
 - [33] I. Shopovska, L. Jovanov, and W. Philips, “RGB-NIR demosaicing using deep residual U-Net,” in *Proceedings of the 2018 26th Telecommunications Forum (TELFOR)*, pp. 1–4, Belgrade, Serbia, November 2018.
 - [34] B. E. Bayer, “Color imaging array,” US Patent No. 3971065, 1976.
 - [35] A. Alsaiani, R. Rustagi, M. M. Thomas, and A. G. Forbes, “Image denoising using a generative adversarial network,” in *Proceedings of the 2019 IEEE 2nd International Conference on Information and Computer Technologies (ICICT)*, pp. 126–132, Kahului, HI, USA, March 2019.
 - [36] 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*, vol. 26, no. 7, pp. 3142–3155, 2017.
 - [37] K. Ma, Z. Duanmu, Q. Wu et al., “Waterloo exploration database: new challenges for image quality assessment models,” *IEEE Transactions on Image Processing*, vol. 26, no. 2, pp. 1004–1016, 2016.
 - [38] D. S. Tan, W.-Y. Chen, and K.-L. Hua, “DeepDemosaicking: adaptive image demosaicking via multiple deep fully convolutional networks,” *IEEE Transactions on Image Processing*, vol. 27, no. 5, pp. 2408–2419, 2018.