Joint Demosaicing and Denoising with Self Guidance

Lin Liu[1,2] Xu Jia[2*] Jianzhuang Liu[2] Qi Tian[2]
[1]CAS Key Laboratory of GIPAS, University of Science and Technology of China
[2]Noah's Ark Lab, Huawei Technologies

Abstract

Usually located at the very early stages of the computational photography pipeline, demosaicing and denoising play important parts in the modern camera image processing. Recently, some neural networks have shown the effectiveness in joint demosaicing and denoising (JDD). Most of them first decompose a Bayer raw image into a four-channel RGGB image and then feed it into a neural network. This practice ignores the fact that the green channels are sampled at a double rate compared to the red and the blue channels. In this paper, we propose a self-guidance network (SGNet), where the green channels are initially estimated and then works as a guidance to recover all missing values in the input image. In addition, as regions of different frequencies suffer different levels of degradation in image restoration. We propose a density-map guidance to help the model deal with a wide range of frequencies. Our model outperforms state-of-the-art joint demosaicing and denoising methods on four public datasets, including two real and two synthetic data sets. Finally, we also verify that our method obtains best results in joint demosaicing , denoising and super-resolution.

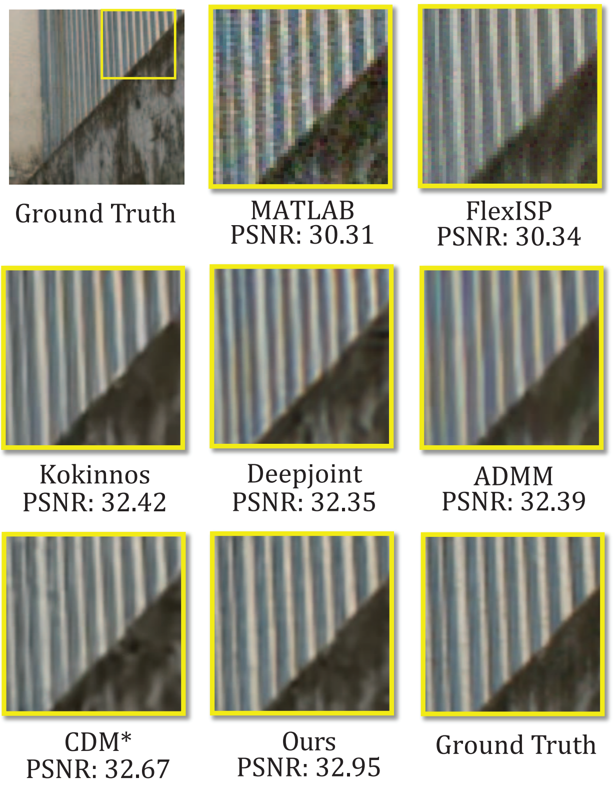

Figure 1: Comparison among six state-of-the-art JDD methods and ours on the MIT moire data set [5]. With the spatially adaptive green-channel guidance, ours removes the color artifacts and recovers the textures significantly.

1. Introduction

Image demosaicing is one of the beginning modules in Image Signal Processing (ISP) pipeline and is a fundamental problem in computer vision. It aims at reconstructing a full-resolution color image from an incomplete observation after a color filter array (CFA) such as the Bayer pattern of RGGB where two-thirds of the image information is missing. Recovering those missing information is an ill-posed problem. In addition, the available one-third RGGB observation is often polluted by different kinds of noise, which further increases the difficulty of the task. These two tasks are important because they appear at the very beginning of the camera imaging pipeline and their performance has crucial influence on the final results. The two tasks of demosaicing and denoising are traditionally processed in a sequential way. However, recent studies show the advantages of the joint approaches [13, 1] .

There has been several approaches proposed for image demosaicing. Demosaicing with simple bilinear interpolation between values on neighboring positions and even deep learning based methods [24, 46, 40, 33, 34] are prone to producing zippering artifacts in regions with edges. To address this issue, a traditional way to alleviate this issue is by applying edge-adaptive interpolation. To further explore

[*]X. Jia is the corresponding author.

useful information in an image to complete the missing pixels, similar to the works on image super-resolution [41, 31], self-similarity within in natural images is explored and exploited to enhance the performance of image demosaicing. However, there are still some visually disturbing artifacts such as moiré patterns appearing on some challenging high frequency regions.

Recently, deep learning methods, which have been shown to be very successful on image recognition tasks [20, 30, 10], also gain popularity on low level vision tasks such as image demosaicing [33, 34, 2, 26]. These methods are able to effectively exploit intra- and inter-channel dependencies in a RAW image to complete missing information. Most of these methods [33, 34, 2] decompose a Bayer raw image into a four-channel RGGB image and feed it into a neural network. They resort to a convolutional neural network itself to discover the relation among the RGGB channels such that they can complement each other to recover the missing pixel values. However, the prior with a Bayer raw image where the red, green and blue chanels are sampled at different rates is not fully exploited. The green channels are sampled at a double rate compared to the red and blue channels. Therefore, making full use of the information from the green channels can be beneficial to the recovery of missing pixel values. There have been several methods [12, 47, 25] proposed before the popularity of deep learning, which first recover the green channels and then recover the RGB channels by exploiting the inter-channel correlation between the green and the red channels as well as that between the green and the blue channels.

In this work, inspired by these methods, we propose a convolutional neural network (CNN) to explicitly explore the guidance within an input Bayer raw image itself for joint demosaicing and denoising. A typical CNN applies the same set of parameters to all positions and all images regardless of specific content within an image. Frequencies and noise vary across images and positions, and therefore, such content-agnostic operation would limit the capacity of a neural network and its capability in addressing demosaicing and denoising tasks. In this work, similar to the traditional methods, we first make an initial estimation based on only the green channels. The initially recovered green channels work as the guidance to conduct spatially adaptive convolution across an image. In this way, the rich information with the green channels is integrated and employed differently at different positions.

In addition, we find that the regions of different frequencies suffer different levels of difficulty in image reconstruction. Knowing which region is more difficult to handle by a model is helpful in the demosaicing and denoising process. This shares some similarity with the noise map in image denoising because challenges vary over different regions and different images. A model without considering frequency difference treats regions of complex patterns and smooth regions equally. The burden of the model is thus severely increased to handle all possible cases, which makes it suboptimal. Estimating a density map for an image and feeding it into a model allow the model to handle a wide range of frequencies. Extensive experiments on both synthetic and real settings demonstrate the effectiveness of the proposed method in addressing the task of joint demosaicing and denoising. Fig. 1 shows a comparison among seven methods.

In summary, our main contributions are:

1. We propose a novel self-guidance network (SGNet) with density-map guidance and green-channel guidance for joint demosaicing and denoising.
2. We propose two losses, adaptive-threshold edge loss and edge-aware smoothness loss, to recover the texture and remove the noise simultaneously.
3. Quantitative and qualitative experimental results on both synthetic and real data sets show that our model outperforms the state-of-the-art methods.

2. Related Work

2.1. Joint Demosaicing and Denoising

Image demosaicing is used to reconstruct a full color image from a sub-sampled output from a color filter array. Demosaicing methods can be divided into traditional methods [40, 17, 25, 26] and deep-learning methods [33, 34, 36]. However, in practical applications, demosaicing is usually not handled independently due to the fact that the Bayer pattern is often corrupted by noise. Some studies try to find a hybrid solution to the problems in camera image processing [51, 4, 48, 22, 28] and obtain good results. Therefore, considering practical situation and benefiting from mixture problem processing, more and more studies begin to handle demosaicing and denoising simultaneously [13, 1]. Joint demosaicing and denoising can get rid of the accumulation of errors after one's processing.

Existing joint demosaicing and denoising methods can also be divided into traditional and deep-learning methods. The former are based on some heuristics, such as total variation minimization [1], sequential energy minimization [18] and learned non-parametric random field [16]. Recently, deep-learning methods have outperformed traditional ones in joint demosaicing and denoising. Gharbi *et al.* [5] trained a deep convolutional neural network on millions of images and achieved state-of-the-art performance. Kokkinos *et al.* [19] proposed an iterative network which combines the majorization-minimization algorithm with a residual denoising network. Ehret *et al.* [3] presented a self-supervised method to demosaicing without the ground-truth. Wronski *et al.* [43] acquired a burst of raw frames and recover a complete RGB image directly from these frames.

2.2. Guided Image Restoration

In many image restoration tasks, a lot of previous works use external information to restore the image. Bilateral filter [39] and some other filters [9, 8] use an external image as guidance to adjust filter parameters, which can reserve sharp edges. They achieve good results in many low-level vision tasks. In recent years, some deep learning methods use guidance information to recover images, especially in the field of super-resolution. In the super-resolution of a depth map, many methods use an RGB image to guide and up-sample the depth map [15, 21, 7]. In order to generate realistic textures of up-sampling images, Wang *et al.* [41] propose to use semantic information to guide the super-resolution of images. Given an image, Zhang *et al.* [49] and Zheng *et al.* [50] introduce style transfer technique and cross-scale warping to super-resolution respectively.

The above methods verify that external guidance information is important for image restoration. However, few studies explore the self-guidance strategy for image restoration. Self-guidance information is often difficult to mine and sometimes requires some prior knowledge. Gu *et al.* [6] propose a self-guidance network for fast image denoising, where the features in the low resolution branch can guide the features in the high resolution branches. In this paper, we propose two self-guidance methods, green channel guidance and density map guidance, specifically for the demosaicing task.

3. The Proposed Method

In this section, we present our SGNet in detail. The architecture of SGNet is shown in Fig. 2.

3.1. Overview

In order to obtain good performance on the task of joint demosaicing and denoising, making full use of the information within an input RGGB raw image is crucial. Here we propose to explore both the green-channel and density information within an input image and use them as guidance for better performance. An input RGGB raw image I^{Bayer}_{RGGB} of size $2H \times 2W$ can be decomposed into four channels, I^{Bayer}_{G1}, I^{Bayer}_{G2}, I^{Bayer}_{R} and I^{Bayer}_{B}, corresponding to the four elements of a Bayer pattern, with each one of size $H \times W \times 1$. We first make an initial estimate of the missing elements for the green channels because there is richer information within the two green channels in the input I^{Bayer}_{G1} and I^{Bayer}_{G2}. The initial estimate of the green channel $\hat{I}_G$ works as a guidance and is applied to the main branch to help recover missing elements for all channels. In addition, we compute a density map M_D (discussed in Sec. 3.2) to represent the difficulty levels of different regions and feed that map as additional input to the main branch network. Finally we present the losses used to train the whole model.

3.2. Density-map guidance

Generally, there are both complex regions with high frequencies and smooth regions with low frequencies in an image. Those regions pose different challenges to the task of joint demosaicing and denoising. Therefore, it is sub-optimal to blindly deal with the whole image in the same way. Inspired by the success of a noise map in image denoising [45], we design a density map to let the network know the difficulty level at each position of the input image. In the density map, regions with dense texture correspond to high frequency patterns and regions with less texture correspond to low frequency patterns. It is computed as:

$$M_D = h\left(g_2\left(I_{gray} - g_1\left(I_{gray}; K_1\right); K_2\right)\right), \tag{1}$$

where both g_1 and g_2 are Gaussian blur operations with kernel sizes K_1 and K_2 respectively, $h(\cdot)$ is a normalization function and I_{gray} is the average of the four decoupled channels, I^{Bayer}_{G1}, I^{Bayer}_{G2}, I^{Bayer}_{R} and I^{Bayer}_{B}. $h(X)$ and I_{gray} are defined as follows:

$$h(X) = \frac{X - \min(X)}{\max(X) - \min(X) + \epsilon} \tag{2}$$

$$I_{gray} = \left(\left(I^{Bayer}_{G1} + I^{Bayer}_{G1}\right)/2 + I^{Bayer}_{R} + I^{Bayer}_{B}\right)/3. \tag{3}$$

After applying a Gaussian blur operation g_1 to an image, regions with dense texture become blurry and difference between the input and output can be very different. Then we apply another Gaussian blur operation g_2 to the difference in order to get a smooth density map. It is further normalized to the range between 0 and 1 with Eq. 2. Fig. 3 shows an example of computing M_D. Once a density map is estimated, we take a simple way to incorporate it by concatenating it to other channels, that is, the four decoupled channels, I^{Bayer}_{G1}, I^{Bayer}_{G2}, I^{Bayer}_{R}, I^{Bayer}_{B} and a noise map I^{noise}. They are combined as the input to the main reconstruction branch. Similar to [5], the noise map I^{noise} denotes the level of the added Gaussian noise during training.

3.3. Green-channel guidance

In a raw image with the Bayer pattern, the number of green pixels is twice that of the red or blue pixels. Thus, it is an easier task to recover the missing elements for the green channel. In addition, usually in an RGB image, the green channel shares similar edges with the red and blue channels; having an initial estimation of the green channel can also benefit the reconstruction of the red and blue channels. As shown in Fig. 2, we first extract the green pixels at the two positions in every 2×2 block in a Bayer image and obtain the two green channels, which together with the

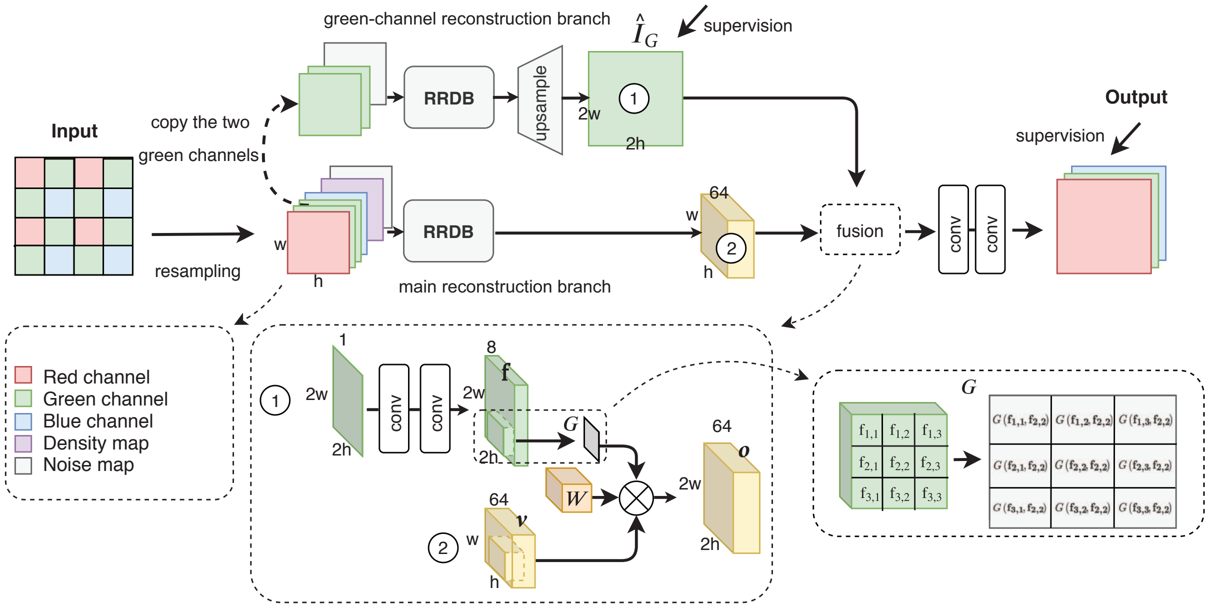

Figure 2: The architecture of the proposed self-guidance network.

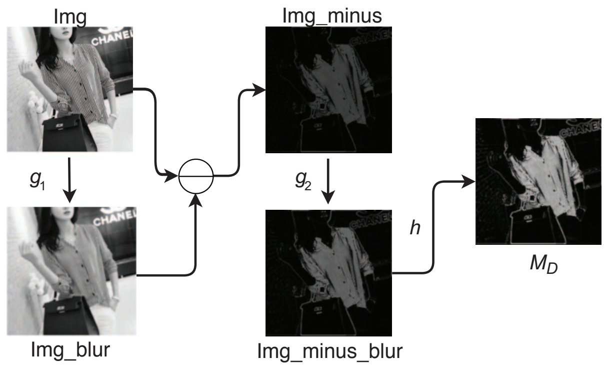

Figure 3: Illustration of computing the density map M_D.

noise map, I^{noise}, are fed to a green-channel reconstruction branch to produce an initial result of the green channel of the output RGB image. The green-channel reconstruction branch is composed of one Residual-in-Residual Dense Block (RRDB) [42]. An RRDB contains multi-level residual connections and dense connections, which allows local information to be sufficiently explored and extracted. The demosaicing task shares some similarity with super-resolution in that both tasks require recovering missing elements and essentially do $2\times$ upsampling. Therefore, similar to the work on image super-resolution [29], we also use a depth-to-space layer at the end of this branch to get a reconstructed green channel $\hat{I}_G$ of size $2H \times 2W \times 1$.

There are several works [12, 46, 36] that employ the initially estimated green channel to reconstruct the rest channels. Most of them simply take $\hat{I}_G$ as an additional input together with the original raw input I^{Bayer}_{RGGB}. By this way, $\hat{I}_G$ is processed with the same set of convolution operations at each position, which may not be optimal. In this work, as shown in Fig. 2, we propose to guide the main reconstruction branch with $\hat{I}_G$ in a novel way. A spatially adaptive convolution operation [32] is applied to the intermediate feature maps in the main reconstruction branch to make the integration process adaptive to the content of the green channel $\hat{I}_G$. Spatially adaptive convolution is content-aware, where the convolution is conducted differently at different positions. It is able to increase the capacity of the network and is helpful in dealing with different frequencies and noise across an image. For the feature of a position j at the intermediate feature maps of the main reconstruction branch $\mathbf{v}_j \in \mathbb{R}^{c'}, j = 1, 2, ..., H \times W$, the output of such spatially adaptive convolution $\mathbf{o}_i \in \mathbb{R}^{c}, i = 1, 2, ..., 2H \times 2W$, is computed as follows:

$$\mathbf{o}_i = \sum_{j \in \Omega(i)} G(\mathbf{f}_i, \mathbf{f}_j)\, \mathbf{W}_{p_{i,j}} \mathbf{v}_j + \mathbf{b}, \tag{4}$$

where $\Omega(\cdot)$ defines an $s \times s$ convolution window, $\mathbf{W} \in \mathbb{R}^{c' \times c \times s \times s}$ is the convolution weights, $p_{i,j}$ is the index of $\mathbf{W}$ related to i and j[1], $\mathbf{b} \in \mathbb{R}^{c'}$ denotes the biases and $\mathbf{f}_i$ is a vector at position i of the feature maps $\mathbf{f}$. $\mathbf{f}$ of size $2H \times 2W \times 8$ is computed from the initially estimated green channel $\hat{I}_G$ with two convolutional layers. $G(\cdot, \cdot)$ is a Gaussian function that depends on the distance between two positions, which is the core to make the convolution operation

[1]The detailed definition of $p_{i,j}$ can be found in [32].

spatially adaptive, and defined as:

$$G(\mathbf{f}_i, \mathbf{f}_j) = \exp\left(-\frac{1}{2}(\mathbf{f}_i - \mathbf{f}_j)^\top (\mathbf{f}_i - \mathbf{f}_j)\right). \quad (5)$$

3.4. Training losses

The whole network model can be trained with a loss for the initial green-channel estimation and a loss for the final RGB image reconstruction. Besides, we introduce two additional losses to further supervise the model training for better demosaicing and denoising performance.

3.4.1 Adaptive-threshold edge loss

Although the network is informed of the per-pixel difficulty level with the additional density map M_D as explained above, each pixel in an input image is supervised with the same strength. However, regions with many high frequency details are more important and should draw more attention than easy-to-recover regions during training. We propose an adaptive-threshold edge loss to address this issue.

We could apply the Canny edge detector to both the output of the network I^O and the target RGB image I^T to extract their edges and obtain two binary edge maps $E(I^O)$ and $E(I^T)$. But the Canny edge detector with a fixed low threshold cannot satisfy every local region in an image. We divide an image into several patches $P_i, i = 1, 2, ..., n$ and find an adaptive threshold for each one. We raise the low threshold for patches with many edges, while reduce the threshold for patches with fewer edges. The low threshold $\theta_i, i = 1, 2, ..., n$, is computed as below:

$$\theta_i = \theta_0 + k\frac{s_{P_i}}{\max(s_{P_1}, s_{P_2}, ..., s_{P_n})}, \theta_0, k > 0, \quad (6)$$

where s_{P_i} is the sum of all pixel values in P_i.

Once the two binary edge maps $E(I^O)$ and $E(I^T)$ are obtained, we can approximate the probability of being edges for a certain patch $p(E(P_i;\theta_i)) \in (0, 1)$, by calculating the proportion of the pixels detected as edge pixels. We then can compute the cross-entropy loss based on the probability $p(E(P_i;\theta_i))$. However, in a natural image, most pixels belong to non-edge regions and only a small portion of the pixels correspond to edges. Similar to [44], we introduce a balancing weight β when calculating the cross-entropy loss, which is shown in Eq. 7.

$$\begin{aligned} L_{edge} = \sum_{i=1}^{n} & -\beta * p(E(P_i^T;\theta_i)) * log(p(E(P_i^O;\theta_i))) \\ & - (1-\beta) * (1 - p(E(P_i^T;\theta_i)))* \\ & log(1 - p(E(P_i^O;\theta_i))), \end{aligned} \quad (7)$$

where β is defined as:

$$\beta = |E(I^T)|/|I^T|, \quad (8)$$

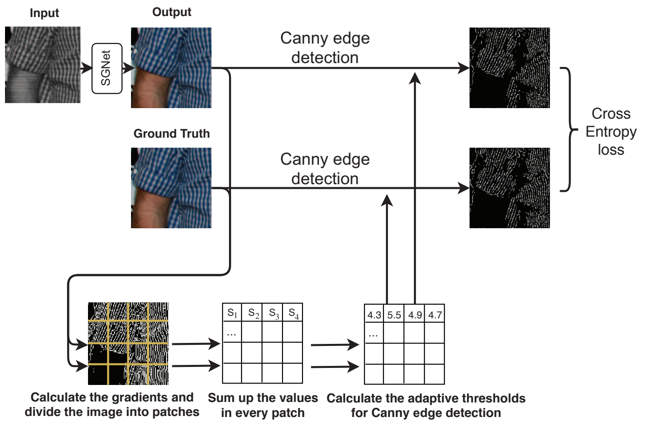
Input
Output
SGNet
Canny edge detection
Ground Truth
Canny edge detection
Cross Entropy loss
s_1 s_2 s_3 s_4
4.3 5.5 4.9 4.7
Calculate the gradients and divide the image into patches
Sum up the values in every patch
Calculate the adaptive thresholds for Canny edge detection

Figure 4: Illustration of computing the adaptive-threshold edge loss.

where $|E(I^T)|$ is the number of edge pixels. With the computed edge loss L_{edge}, regions with strong edges are strictly supervised such that small errors incur larger punishment. The procedure to compute L_{edge} is illustrated in Fig. 4.

3.4.2 Edge-aware smoothness loss

In order to achieve good performance in denoising, the total variation (TV) regularization is used to smooth out noise and unexpected artifacts. However, the TV loss fails at regions full of textual patterns. Thus, we propose an edge-aware smoothness loss L_{smooth}, which is formulated as:

$$L_{smooth} = \|\nabla O \circ \exp(-\lambda \nabla O)\|, \quad (9)$$

where $\nabla O = \|\nabla O_h\| + \|\nabla O_v\|$ is the sum of the gradients in the horizontal and vertical directions, and λ is a parameter balancing the strength of edge-awareness. This loss is the TV loss multiplied by an exponential smoothing term. In smooth regions, it is relatively larger than in regions with rich texture and edges. With such an edge-aware smoothness loss, the model learns to simultaneously get rid of noise in smooth regions while keeping edges in textured regions.

Finally, the overall loss to train the whole network is defined as follows:

$$L = L_{edge} + \lambda_1 L_{smooth} + \lambda_2 L_{l_1} + \lambda_3 L_g, \quad (10)$$

where L_{l_1} is the l_1 loss between the output image and its ground-truth, and L_g is the l_1 loss between $\hat{I}_G$ and the green channel of the ground-truth.

4. Experiments

Extensive experiments on both real and synthetic datasets show that our method outperforms state-of-the-arts with large margins. Ablation study is also conducted to analyze the importance of the density-map guidance and the green-channel guidance.

Method	σ	Dense texture		Sparse texture		MIT moire			Urban100		
		PSNR	LPIPS	PSNR	LPIPS	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS
FlexISP [11]	5	36.64	–	38.82	–	29.06	0.8206	–	30.37	0.8832	–
SEM [18]		38.55	–	38.40	–	27.46	0.8292	–	27.19	0.7813	–
ADMM [35]		39.97	–	43.78	–	28.58	0.7923	–	28.57	0.8578	–
Deepjoint [5]		45.90	0.01782	47.48	0.02218	31.82	0.9015	0.0692	34.04	0.9510	0.0304
Kokkinos [19]		44.77	0.02308	47.40	0.02158	31.94	0.8882	0.0711	34.07	0.9358	0.0437
CDM* [36]		45.49	0.02050	47.24	0.02281	30.36	0.8807	0.0757	32.09	0.9311	0.0484
SGNet		**46.75**	**0.01433**	**47.88**	**0.01917**	**32.15**	**0.9043**	**0.0691**	**34.54**	**0.9533**	**0.0299**
FlexISP [11]	10	32.66	–	33.62	–	26.61	0.7491	–	27.51	0.8196	–
SEM [18]		31.26	–	31.17	–	25.45	0.7531	–	25.36	0.7094	–
ADMM [35]		37.64	–	40.25	–	28.26	0.7720	–	27.48	0.8388	–
Deepjoint [5]		43.26	0.04017	45.17	0.04139	29.75	0.8561	0.1066	31.60	0.9152	0.0610
Kokkinos [19]		41.95	0.05499	44.86	0.04328	30.01	0.8123	0.1132	31.73	0.8912	0.0710
CDM* [36]		42.46	0.05393	44.70	0.04535	28.63	0.8286	0.1304	30.03	0.8934	0.0832
SGNet		**44.23**	**0.03280**	**45.56**	**0.03764**	**30.09**	**0.8619**	**0.1034**	**32.14**	**0.9229**	**0.0546**
FlexISP [11]	15	29.67	–	30.48	–	24.91	0.6851	–	25.55	0.7642	–
SEM [18]		25.98	–	27.01	–	23.23	0.6527	–	23.25	0.6156	–
ADMM [35]		34.87	–	36.78	–	27.58	0.7497	–	28.37	0.8440	–
Deepjoint [5]		41.42	0.06447	43.54	0.05783	28.22	0.8088	0.1506	29.73	0.8802	0.0929
Kokkinos [19]		40.16	0.08046	43.11	0.06103	28.28	0.7693	0.1764	29.87	0.8451	0.1054
CDM* [36]		40.51	0.08306	42.98	0.06549	27.23	0.7775	0.1875	28.34	0.8543	0.1262
SGNet		**42.32**	**0.05491**	**43.94**	**0.05456**	**28.60**	**0.8188**	**0.1412**	**30.37**	**0.8923**	**0.0793**

Table 1: Quantitative comparison for joint demosaicing and denoising on real test sets (Dense texture and Sparse texture) and synthetic test sets (MIT moire and Urban100). PSNR, SSIM: higher is better; LPIPS: lower is better.

	Dense texture			Sparse texture		
σ	5	10	15	5	10	15
Metrics	PSNR/LPIPS	PSNR/LPIPS	PSNR/LPIPS	PSNR/LPIPS	PSNR/LPIPS	PSNR/LPIPS
w/o green branch	45.43/0.02373	42.90/0.04381	41.21/0.06654	47.15/0.01995	44.30/0.03935	42.44/0.05953
Concatenation guidance	45.93/0.02240	43.31/0.04182	42.05/0.06006	47.46/0.01885	45.16/0.03846	43.75/0.05782
Adaptive guidance	**46.75/0.01433**	**44.23/0.03280**	**42.32/0.05491**	**47.88/0.01917**	**45.56/ 0.03764**	**43.94/0.05456**

Table 2: Ablation study of the green-channel guidance method on Pixelshift200.

4.1. Datasets and Implementation Detail

Datasets. For the synthetic datasets, we combine DIV2K [38] (containing 800 2K-resolution images) and Flickr2K [37] (containing 2650 2K-resolution images) as the training set. We choose the MIT moire [5] and Urban100 [14] as the test sets. MIT moire has 1000 128×128-resolution hard-case images. Urban100 contains 100 high-resolution images. For the real datasets, we choose Pixelshift200 [27] for training, and conduct test experiments on Dense texture [27] and Sparse texture [27]. The images in Dense texture are selected by us using the metrics proposed in [5]. Sparse texture is the original test set in Pixelshift200. All the images are randomly cropped into patches of size 128×128 and perturbed by Gaussian noise with $\sigma \in [0, 16]$.

Implementation detail. For all experiments, our model is implemented in Pytorch and runs on a NVIDIA Tesla V100 GPU. The batch size is set to 8 during training. We use Adam with $\beta_1 = 0.9$ and $\beta_2 = 0.999$ for optimization. The learning rate is 0.0001 and reduced to one-tenth for every 200,000 epochs. In Eq. 10, the parameters λ_1 and λ_2 are set to 5 and 50 respectively. $\lambda_3 = 30 - n_{iter} * 27/700000$, where n_{iter} is the number of training iterations. λ_3 is defined like this because the green channel reconstruction is given more emphasis at the early stage of training and less emphasis at later. θ_0 and k in Eq. 6 are 1 and 4 respectively. We assume JDD is done before white balance. We follow most related works [27, 19, 5] and do not use inverse ISP to pre-process sRGB images.

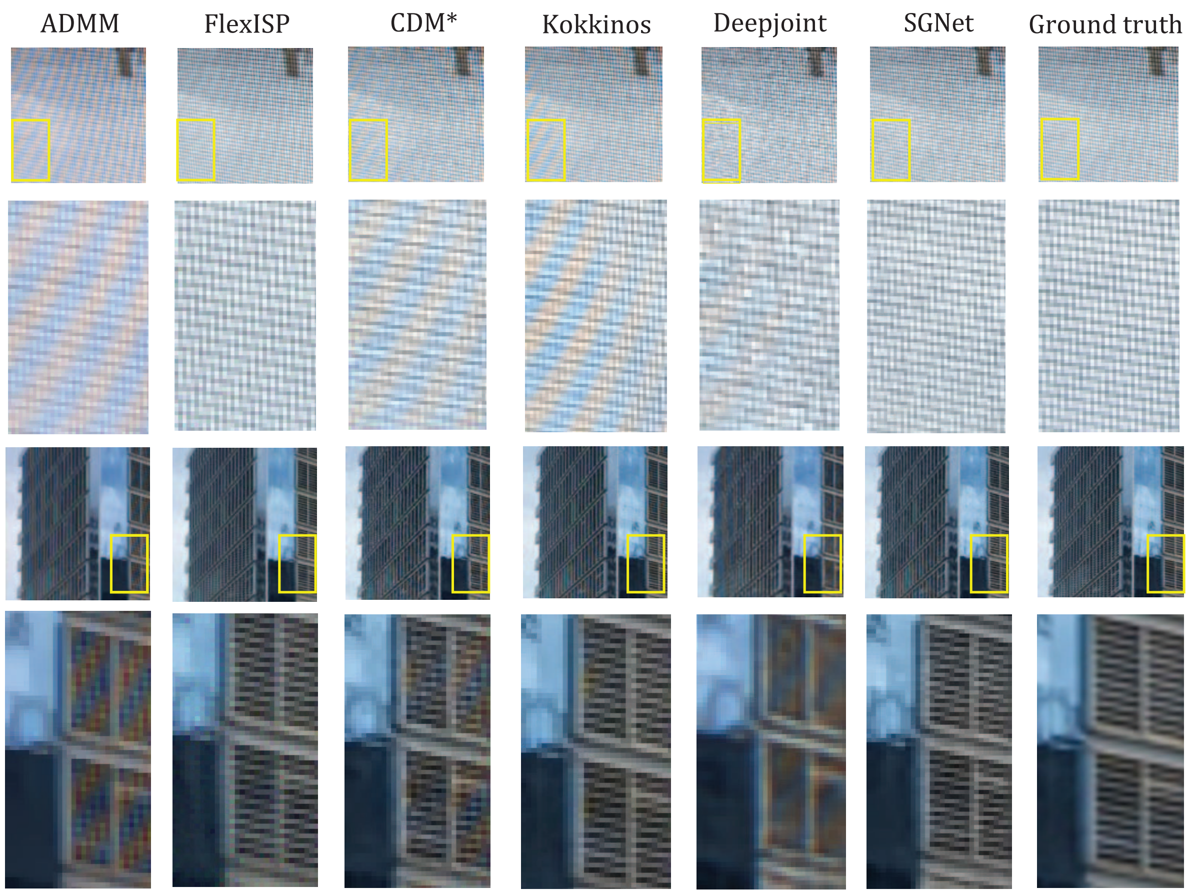

Figure 5: Visual comparison between state-of-the-arts and our method for joint demosaicing and denoising.

Method	σ	Dense texture PSNR/LPIPS	Sparse texture PSNR/LPIPS
w/o L_{smooth}		44.01/0.03574	45.40/0.03933
w/o L_{edge}	10	43.92/0.03752	45.42/0.03905
w/o M_D		43.86/0.04073	45.37/0.03982
Full model		**44.23/0.03280**	**45.56/0.03764**

Table 3: Ablation study on Pixelshift200.

Algorithm	Deepjoint	CDM	Kokkinos	Ours
Time (ms/frame)	86	64	290	56
Parameter (MB)	0.50	0.64	–	0.71

Table 4: Efficiency comparison on Dense texture.

4.2. Ablation study

4.2.1 Green-Channel Guidance

To verify the contributions of the green-channel guidance with spatially adaptive convolution, we conduct an ablation study on Dense texture and Sparse texture of Pixelshift200. In the first column of Table 2, “adaptive guidance” means our full model; “Concatenation guidance” means we concatenate $\hat{I}_G$ with the output feature maps of the RRDB in the main reconstruction branch. As shown in Table 2, we can see that “Concatenation guidance” and “Adaptive guidance” improve the PSNR on both test sets. The latter obtains better results than the former. “Adaptive guidance” is content-aware and has the advantage of handling different frequencies and noise across an image.

4.2.2 Component Analysis

To verify the contributions of L_{smooth}, L_{edge} and M_D of SGNet, we conduct an ablation study on Dense texture and Sparse texture with the noise level of 10. As shown in Table 3, without L_{smooth}, L_{edge} or M_D, the PSNR decreases by 0.22dB, 0.31dB or 0.37dB in Dense texture, respectively. M_D informs the network the difficulty level at each position and is more important than L_{smooth} and L_{edge}. L_{smooth} and L_{edge} help to remove noise and recover textures, respectively.

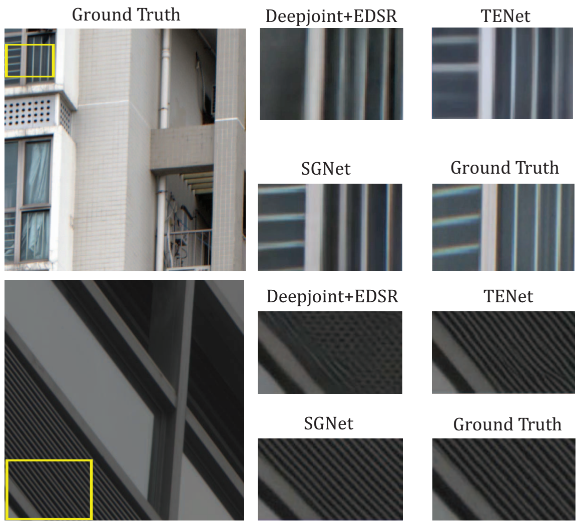

Figure 6: Visual comparison in the task of joint demosaicing, denoising and super-resolution on Pixelshift200.

4.3. Comparison with State-of-the-Art

We compare our method with state-of-the-art methods, including 3 traditional methods (FlexISP [11], SEM [18], ADMM [35]) and 3 deep learning methods (Deepjoint [5], CDM [36], Kokkinos [19]). We use their source code for comparison. For all methods, the Gaussian noise level is known. The noise level map is used as an additional input along with the original input in CDM. Thus, we denote the altered model as CDM*. ADMM and FlexISP iterate 40 and 30 times respectively for each input image.

4.4. Quantitative Evaluation

For quantitative evaluation, we choose 3 evaluation metrics, the standard Peak Signal To Noise Ratio (PSNR), Structural Similarity (SSIM), and Learned Perceptual Image Patch Similarity (LPIPS) [48] which measures perceptual image similarity using a pre-trained deep network. Because the pre-trained model of LPIPS is based on Python and the code for the traditional methods are based on MATLAB, LPIPS is not used to evaluate their performance. In Table 1, we conduct 4 experiments on 3 different noise levels ($\sigma = 5, 10, 15$). As shown in Table 1, our method outperforms the state-of-the-art methods on joint demosaicing and denoising.

Because traditional methods run much longer than deep-learning methods, we only evaluate the efficiency of all the deep-learning JDD methods on one NVIDIA Tesla V100 GPU. As shown in Table 4, although the number of parameters in our model is slightly more than other methods, ours is the fastest among all the 4 deep learning methods.

Method	σ	Dense texture PSNR/LPIPS	Sparse texture PSNR/LPIPS
Deepjoint [5]+EDSR [23]	5	44.26/0.03197	46.18/0.03667
TENet [27]	5	44.78/0.03011	**46.25/0.03530**
SGNet	5	**45.93/0.02578**	46.16/0.03578
Deepjoint [5]+EDSR [23]	10	41.84/0.05988	43.67/0.06232
TENet [27]	10	42.70/0.04886	43.86/0.05830
SGNet	10	**43.81/0.04023**	**43.91/0.05551**
Deepjoint [5]+EDSR [23]	15	40.17/0.08588	42.06/0.08003
TENet [27]	15	41.06/0.07032	42.25/0.07592
SGNet	15	**42.23/0.06029**	**42.33/0.07190**

Table 5: Quantitative comparison for JDDS on Pixelshift200.

4.5. Qualitative Results

As shown in Fig. 5, the results of ADMM, CDM*, Kokkinos and Deepjoint have moiré artifacts. Though FlexISP produced better results there is still noise left. In addition, ADMM and Deepjoint tend to blur the high-frequency regions (see the 2nd image in Fig. 5). In contrast, our SGNet eliminates the color moiré artifacts more effectively, benefiting from the green-channel guidance. In addition, our model restores some aliasing textures and remove the noise in high-frequency regions, mainly thanks to M_D, L_{edge} and L_{smooth}. Due to the space limitation, we show some results on Dense texture in the supplementary material.

5. Joint Demosaicing, Denoising and Super-resolution

To further verify the effectiveness of our method, we conduct an experiment on the mixture problem of joint demosaicing, denoising and super-resolution (JDDS). Qian et.al. [27] shows that handling these three tasks together can achieve better results for camera signal processing. We train SGNet and test it on the corresponding test set. The results are shown in Table 5 and Fig. 6. Our model outperforms the hybrid solution model TENet and the sequential solution method Deepjoint+EDSR.

6. Conclusion

We have proposed a novel self-guidance network (SGNet) for joint demosaicing and denoising. Our spatially adaptive guidance with green-channel is content-aware. SGNet removes color artifacts and retains image details simultaneously. Our method outperforms other state-of-the-art JDD models significantly. Besides, SGNet obtains the best result in joint demosaicing, denoising and super-resolution. In future, we will explore more prior information with the task to improve the performance of joint demosaicing and denoising.

References

[1] Laurent Condat and Saleh Mosaddegh. Joint demosaicking and denoising by total variation minimization. In *ICIP, 2012*. 1, 2

[2] Weishong Dong, Ming Yuan, Xin Li, and Guangming Shi. Joint demosaicing and denoising with perceptual optimization on a generative adversarial network. *arXiv preprint arXiv:1802.04723*, 2018. 2

[3] Thibaud Ehret, Axel Davy, Pablo Arias, and Gabriele Facciolo. Joint demosaicing and denoising by overfitting of bursts of raw images. *arXiv preprint arXiv:1905.05092*, 2019. 2

[4] Sina Farsiu, Michael Elad, and Peyman Milanfar. Multi-frame demosaicing and super-resolution from undersampled color images. In *TCI II, 2004*. 2

[5] Michaël Gharbi, Gaurav Chaurasia, Sylvain Paris, and Frédo Durand. Deep joint demosaicking and denoising. *TOG, 2016*. 1, 2, 3, 6, 8

[6] Shuhang Gu, Yawei Li, Luc Van Gool, and Radu Timofte. Self-guided network for fast image denoising. In *ICCV, 2019*. 3

[7] Shuhang Gu, Wangmeng Zuo, Shi Guo, Yunjin Chen, Chongyu Chen, and Lei Zhang. Learning dynamic guidance for depth image enhancement. In *CVPR, 2017*. 3

[8] Bumsub Ham, Minsu Cho, and Jean Ponce. Robust image filtering using joint static and dynamic guidance. In *CVPR, 2015*. 3

[9] Kaiming He, Jian Sun, and Xiaoou Tang. Guided image filtering. *TPAMI, 2012*. 3

[10] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR, 2016*. 2

[11] Felix Heide, Markus Steinberger, Yun-Ta Tsai, Mushfiqur Rouf, Dawid Pajak, Dikpal Reddy, Orazio Gallo, Jing Liu, Wolfgang Heidrich, Karen Egiazarian, et al. Flexisp: A flexible camera image processing framework. In *TOG, 2014*. 6, 8

[12] Keigo Hirakawa and Thomas W Parks. Adaptive homogeneity-directed demosaicing algorithm. *TIP, 2005*. 2, 4

[13] Keigo Hirakawa and Thomas W Parks. Joint demosaicing and denoising. *TIP, 2006*. 1, 2

[14] Jia-Bin Huang, Abhishek Singh, and Narendra Ahuja. Single image super-resolution from transformed self-exemplars. In *CVPRW, 2015*. 6

[15] Tak-Wai Hui, Chen Change Loy, and Xiaoou Tang. Depth map super-resolution by deep multi-scale guidance. In *ECCV, 2016*. 3

[16] Daniel Khashabi, Sebastian Nowozin, Jeremy Jancsary, and Andrew W Fitzgibbon. Joint demosaicing and denoising via learned nonparametric random fields. *TIP, 2014*. 2

[17] Daisuke Kiku, Yusuke Monno, Masayuki Tanaka, and Masatoshi Okutomi. Residual interpolation for color image demosaicking. In *ICIP, 2013*. 2

[18] Teresa Klatzer, Kerstin Hammernik, Patrick Knobelreiter, and Thomas Pock. Learning joint demosaicing and denoising based on sequential energy minimization. In *ICCP, 2016*. 2, 6, 8

[19] Filippos Kokkinos and Stamatios Lefkimmiatis. Deep image demosaicking using a cascade of convolutional residual denoising networks. In *ECCV, 2018*. 2, 6, 8

[20] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS, 2012*. 2

[21] Yijun Li, Jia-Bin Huang, Narendra Ahuja, and Ming-Hsuan Yang. Deep joint image filtering. In *ECCV, 2016*. 3

[22] Zhetong Liang, Jianrui Cai, Zisheng Cao, and Lei Zhang. Cameranet: A two-stage framework for effective camera isp learning. *arXiv preprint arXiv:1908.01481*, 2019. 2

[23] Bee Lim, Sanghyun Son, Heewon Kim, Seungjun Nah, and Kyoung Mu Lee. Enhanced deep residual networks for single image super-resolution. In *CVPRW, 2017*. 8

[24] Henrique S Malvar, Li-wei He, and Ross Cutler. High-quality linear interpolation for demosaicing of bayer-patterned color images. In *ICASSP, 2004*. 1

[25] Yusuke Monno, Daisuke Kiku, Masayuki Tanaka, and Masatoshi Okutomi. Adaptive residual interpolation for color image demosaicking. In *ICIP, 2015*. 2

[26] Yan Niu, Jihong Ouyang, Wanli Zuo, and Fuxin Wang. Low cost edge sensing for high quality demosaicking. *TIP, 2018*. 2

[27] Guocheng Qian, Jinjin Gu, Jimmy S Ren, Chao Dong, Furong Zhao, and Juan Lin. Trinity of pixel enhancement: a joint solution for demosaicking, denoising and super-resolution. *arXiv preprint arXiv:1905.02538*, 2019. 6, 8

[28] Sivalogeswaran Ratnasingam. Deep camera: A fully convolutional neural network for image signal processing. *arXiv preprint arXiv:1908.09191*, 2019. 2

[29] Wenzhe Shi, Jose Caballero, Ferenc Huszár, Johannes Totz, Andrew P Aitken, Rob Bishop, Daniel Rueckert, and Zehan Wang. Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network. In *CVPR, 2016*. 4

[30] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. 2

[31] Jae Woong Soh, Gu Yong Park, Junho Jo, and Nam Ik Cho. Natural and realistic single image super-resolution with explicit natural manifold discrimination. In *CVPR, 2019*. 2

[32] Hang Su, Varun Jampani, Deqing Sun, Orazio Gallo, Erik Learned-Miller, and Jan Kautz. Pixel-adaptive convolutional neural networks. In *CVPR, 2019*. 4

[33] Nai-Sheng Syu, Yu-Sheng Chen, and Yung-Yu Chuang. Learning deep convolutional networks for demosaicing. *arXiv preprint arXiv:1802.03769*, 2018. 1, 2

[34] Daniel Stanley Tan, Wei-Yang Chen, and Kai-Lung Hua. Deepdemosaicking: Adaptive image demosaicking via multiple deep fully convolutional networks. *TIP, 2018*. 1, 2

[35] Hanlin Tan, Xiangrong Zeng, Shiming Lai, Yu Liu, and Maojun Zhang. Joint demosaicing and denoising of noisy bayer images with admm. In *ICIP, 2017*. 6, 8

[36] Runjie Tan, Kai Zhang, Wangmeng Zuo, and Lei Zhang. Color image demosaicking via deep residual learning. In *ICME, 2017*. 2, 4, 6, 8

[37] Radu Timofte, Eirikur Agustsson, Luc Van Gool, Ming-Hsuan Yang, and Lei Zhang. Ntire 2017 challenge on single image super-resolution: Methods and results. In *CVPRW, 2017*. 6

[38] Radu Timofte, Shuhang Gu, Jiqing Wu, Luc Van Gool, Lei Zhang, Ming-Hsuan Yang, Muhammad Haris, et al. Ntire 2018 challenge on single image super-resolution: Methods and results. In *CVPRW, 2018*. 6

[39] Carlo Tomasi and Roberto Manduchi. Bilateral filtering for gray and color images. In *ICCV, 1998*. 3

[40] Chi-Yi Tsai and Kai-Tai Song. A new edge-adaptive demosaicing algorithm for color filter arrays. *IVC, 2017*. 1, 2

[41] Xintao Wang, Ke Yu, Chao Dong, and Chen Change Loy. Recovering realistic texture in image super-resolution by deep spatial feature transform. In *CVPR, 2018*. 2, 3

[42] Xintao Wang, Ke Yu, Shixiang Wu, Jinjin Gu, Yihao Liu, Chao Dong, Yu Qiao, and Chen Change Loy. Esrgan: Enhanced super-resolution generative adversarial networks. In *ECCV, 2018*. 4

[43] Bartlomiej Wronski, Ignacio Garcia-Dorado, Manfred Ernst, Damien Kelly, Michael Krainin, Chia-Kai Liang, Marc Levoy, and Peyman Milanfar. Handheld multi-frame super-resolution. *arXiv preprint arXiv:1905.03277*, 2019. 2

[44] Saining Xie and Zhuowen Tu. Holistically-nested edge detection. In *ICCV, 2015*. 5

[45] Kai Zhang, Wangmeng Zuo, and Lei Zhang. Ffdnet: Toward a fast and flexible solution for CNN based image denoising. *TIP, 2018*. 3

[46] Lei Zhang and Xiaolin Wu. Color demosaicking via directional linear minimum mean square-error estimation. *TIP, 2005*. 1, 4

[47] Lei Zhang, Xiaolin Wu, Antoni Buades, and Xin Li. Color demosaicking by local directional interpolation and nonlocal adaptive thresholding. *JEI, 2011*. 2

[48] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *CVPR, 2018*. 2, 8

[49] Zhifei Zhang, Zhaowen Wang, Zhe Lin, and Hairong Qi. Image super-resolution by neural texture transfer. In *CVPR, 2019*. 3

[50] Haitian Zheng, Mengqi Ji, Haoqian Wang, Yebin Liu, and Lu Fang. Crossnet: An end-to-end reference-based super resolution network using cross-scale warping. In *ECCV, 2018*. 3

[51] Ruofan Zhou, Radhakrishna Achanta, and Sabine Süsstrunk. Deep residual network for joint demosaicing and super-resolution. In *CIC, 2018*. 2