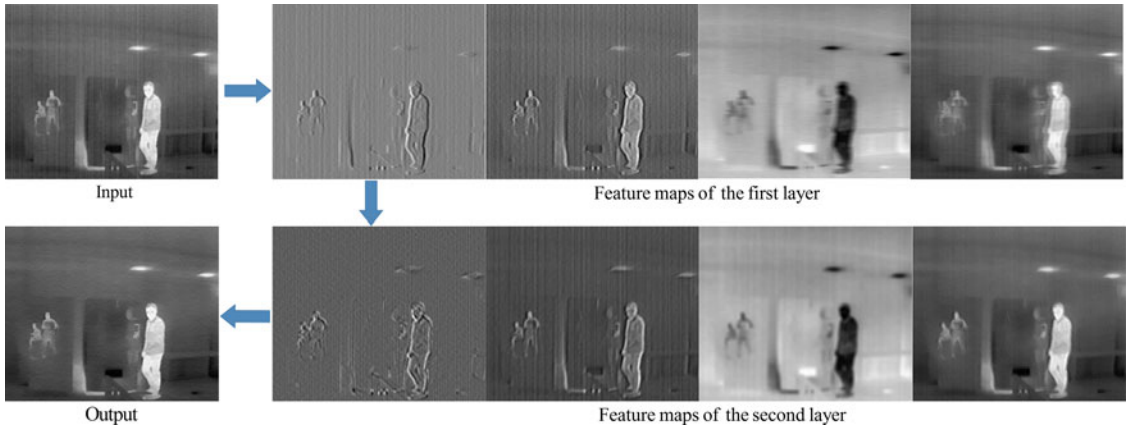
IEEE Photonics Journal
An IEEE Photonics Society Publication

Open Access

Single Infrared Image Stripe Noise Removal Using Deep Convolutional Networks

Volume 9, Number 4, August 2017

Xiaodong Kuang
Xiubao Sui
Qian Chen
Guohua Gu



DOI: 10.1109/JPHOT.2017.2717948
1943-0655 © 2017 IEEE

IEEE photonics SOCIETY
IEEE

Single Infrared Image Stripe Noise Removal Using Deep Convolutional Networks

Xiaodong Kuang, Xiubao Sui, Qian Chen, and Guohua Gu

Nanjing University of Science and Technology, Nanjing 210094, China

DOI:10.1109/JPHOT.2017.2717948
1943-0655 © 2017 IEEE. Translations and content mining are permitted for academic research only.
Personal use is also permitted, but republication/redistribution requires IEEE permission.
See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

Manuscript received March 3, 2017; revised June 12, 2017; accepted June 16, 2017. Date of publication June 21, 2017; date of current version July 3, 2017. This work was supported in part by the National Natural Science Foundation of China under Grant 11503010, in part by the Fundamental Research Funds for the Central Universities under Grant 30916015103, in part by Qing Lan Project and Open Research Fund of Jiangsu Key Laboratory of Spectral Imaging & Intelligence Sense under Grant 3091601410405, and in part by Zijin Intelligent Program, Nanjing University of Science and Technology.
Corresponding author: Xiubao Sui (e-mail: sxbhandsome@163.com).

Abstract: In this paper, we present a deep learning method for single infrared image stripe noise removal. Our method is denoted as a deep convolutional neural network (CNN) that takes the noisy image as the input and outputs the clean image. The deep CNN consists of two components: 1) image denoising, substantially removing the stripe noise but losing details, 2) image denoising and super resolution, completely eliminating the residual stripe noise and restore details. Our deep CNN exhibits excellent image denoising and detail preserving performance. Meanwhile it achieves fast speed for real-time image processing. Experiments study the relationship between model parameter settings and model performance.

Index Terms: Stripe noise removal, infrared image, deep convolutional neural networks.

1. Introduction

The stripe noise (SN), which gradually appears as ambient temperature changes and the camera's internal temperature increases, is an urgent problem [1]. In order to solve the problem, researchers have proposed two types of methods. One type is calibration-based method [2], [3], which interrupts the image operation and affects user's observation of the target, especially in military applications. Another type is scene-based method [4]–[9], which has some limitation conditions, such as the high-precision registration and the random scene. Therefore, removing the stripe noise in a single frame by strong prior information is currently the main research direction. Recent state-of-the-art methods mostly exploit a prior information existing between adjacent columns of the image.

The midway histogram equalization (MHE) is one of the representative stripe noise removal (SNR) methods [10]. This method assumes that single columns carry enough information by themselves. The images being continuous, the difference between two adjacent columns is statistically small, implying that two neighboring histograms are nearly equal. So the histogram of each column can be transported to the midway of histograms of neighboring columns. This method works well for removing the slight stripe noise, but cannot remove the stripe noise which forms low frequency noise on the space. For instance, due to slow transformation in the time domain, stripe noise makes up the low frequency noise in the spatial domain. When we compute a new cumulative

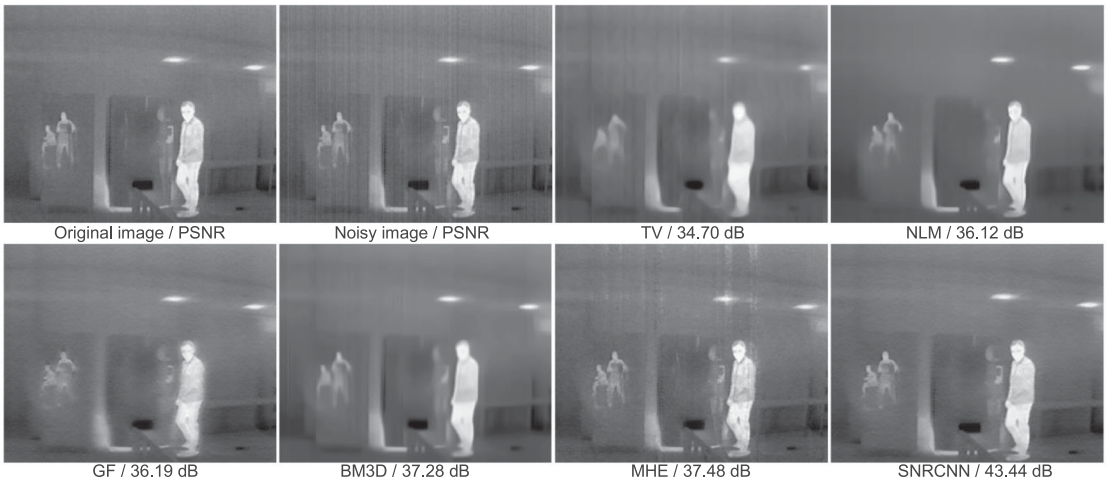


Fig. 1. The stripe noise added on the original image is a Gaussian distribution with zero mean and standard deviation of 0.01. Compared with other state-of-the-art methods, our method shows visually appealing reconstructed quality.

histogram for a column in the area of the low-frequency noise, the adjacent columns are always in a low-frequency area that makes no contribution to the stripe noise removal method. Therefore, the low-frequency stripe noise is retained. In addition, the histogram of each column is replaced by the midway histogram of the neighboring columns. Therefore, to a certain extent, it reduces the spatial resolution of the image and is prone to emerge ghost artifacts (see Fig. 1).

Similar to MHE, Total variation (TV) [11] is to add the constant to each column of the image and then minimizes the difference between two adjacent columns to optimize the constant. They exploit the information between adjacent columns to filter out the noise and their operation on the images can be considered as the row convolution kernels. Meanwhile, the method in [12] uses a convolutional neural network including some square convolution kernels to effectively restore image detail. Inspired by this fact, we consider a convolutional neural network containing image denoising (using row convolution kernels), image denoising and super resolution (using square convolution kernels). Our approach differs fundamentally from current existing methods, in that ours does not need to learn any prior information in advance and preserves details. These prior information is automatically obtained by learning.

We name the proposed model Stripe Noise Removal Convolutional Neural Network (SNRCNN[1]). The proposed SNRCNN has some attractive properties. Firstly, our method produces superior restoration quality. Fig. 1 shows an example of a comparison with other state-of-the-art methods. Secondly, our approach achieves fast speed for real-time image processing. Compared with other methods, our method does not need to optimize the parameters, which have already been obtained through learning. Thirdly, experiments suggest that the network performance may be further boosted using more training samples, and a larger and deeper model.

In general, the contributions of this paper are as follows:

1) We propose a fully convolutional neural network for stripe noise removal. To our knowledge, it is the first to use a deep learning method to eliminate the stripe noise.
2) We pioneer the combination of image denoising and SR for stripe noise removal. Compared with the state-of-the-art methods, our method can greatly preserve details.
3) We show that deep learning can be used to solve the traditional problem of stripe noise. Compared with the state-of-the-art methods, our method achieves superior restoration quality and speed.

[1]The implementation is available at https://github.com/Kuangxd/SNRCNN_Matlab and https://github.com/Kuangxd/Train_SNRCNN

The remainder of this paper is as follows. Section 2 briefly introduces the related work. Section 3 presents the deep convolutional neural network for stripe noise removal. Experiments in Section 4 study the relationship between model parameter settings and model performance. Section 5 concludes.

2. Related Work

2.1 Image Denoising Methods

Currently, image denoising methods mainly exploit the self-similarity property of the pixels in images. Five most representative methods are Total Variation (TV), MHE, Guided Filter (GF) [13], Non-Local Means (NLM) [14] and Block Method of 3-Dimension (BM3D) [15].

TV method considers that the energy function of each column is very small and the adjacent columns have the same pixel value. MHE method assumes that single columns carry enough information by themselves and two neighboring histograms are nearly equal. GF method takes the raw data as its input and guidance images, and smooths similar pixels within a row window. NLM reduces the noise of zero-mean Gaussian white noise by weighted averaging these self-similar blocks to estimate the center of the reference block. Although NLM has achieved a great denoising effect, the protection for the structural information in original images is still not enough and the time complexity is relatively high. In [15], a three-dimensional block matching algorithm is proposed based on the similarity between image blocks. BM3D supposes that patches in a stack are variants of a same patch, and thus the variations between these transformed patches and the ideal patch (in the transformed domain) should also be sparse. The author in [17] propose an algorithm based on grayscale succession to eliminate stripe noise in infrared images. The method utilizes a local window and grayscale succession between the adjacent row pixels to obtain the parameters, which can be more accurate by setting adaptive threshold. However, the setting of the threshold depends on experience and may be reset for different infrared images.

However, in the infrared field, the details of the infrared image are weaker than that of the visible image. When we use TV, GF, NLM and BM3D methods to remove the stripe noise in the infrared images, it is easy to lose the structure information in the images.

2.2 Deep Convolutional Neural Network for Stripe Noise Removal

Deep CNNs have recently shown a huge research hotspot due to its excellent performance in image processing, such as image classification [17], [18], target detection [19]–[21], face recognition [22] and super resolution [13]. Through deep learning, the network automatically learns all kinds of prior information of images and saves them in hidden layers. One of the most representative applications is image SR using deep convolutional networks [13]. Our method is also inspired by this successful implementation and uses it to cope with the problem of losing details in stripe noise removal.

3. Neural Network for Stripe Noise Removal

3.1 Formulation

Consider a single clean normalized image $\boldsymbol{X}$, we first add a stripe noise of a Gaussian distribution with zero mean and standard deviation of 0.02 to obtain a noisy image. The noisy image is defined as $\boldsymbol{Y}$. Our purpose is to get from $\boldsymbol{Y}$ the image M($\boldsymbol{Y}$) that removes the stripe noise and preserve details. The function M theoretically consists of two operations:

1) *Image denoising:* this operation uses a layer to filter the image $\boldsymbol{Y}$. Note that the convolution kernels in this layer are row vectors. After filtering, we get a set of feature maps, of which the stripe noise and the solution are greatly reduced.
2) *Image denoising and super resolution:* this operation uses two layers to process the above feature maps. Note that the convolution kernels in both layers are square. The first layer nonlinearly extracts patches from the feature maps and generate another set of feature maps,

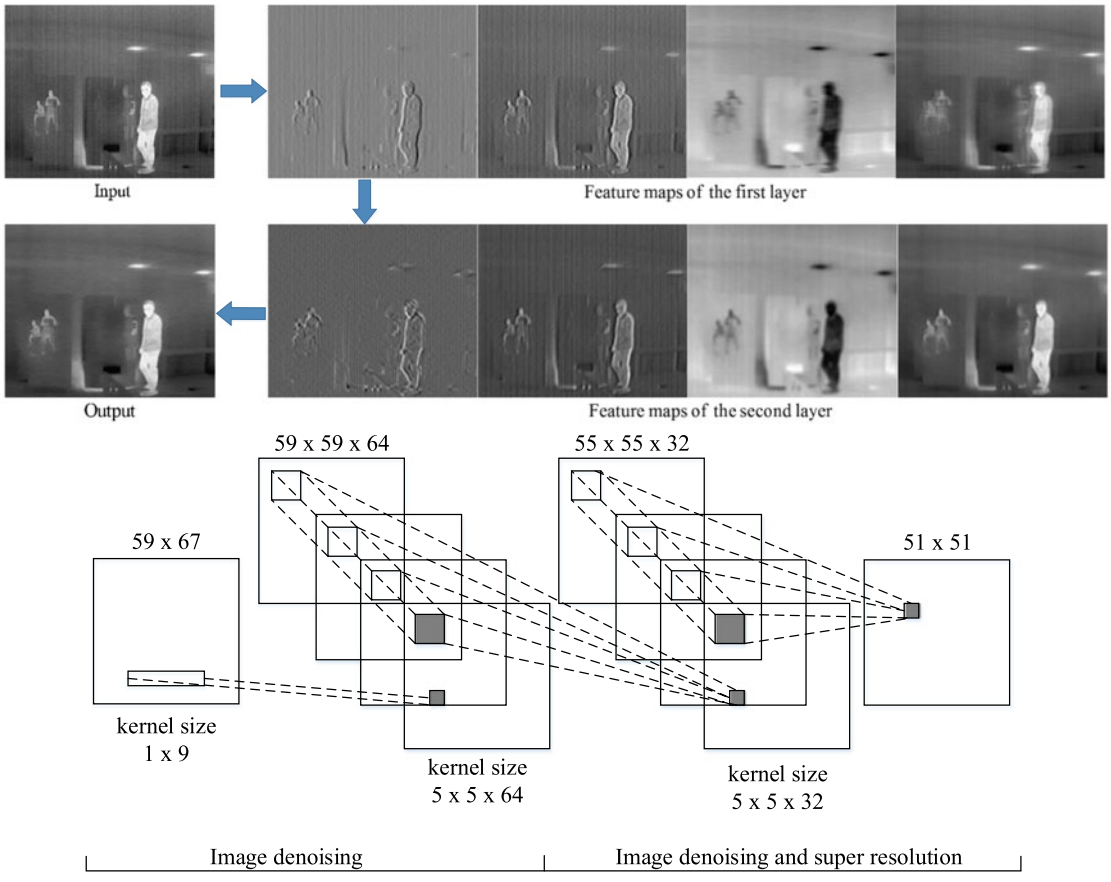


Fig. 2. Our complete network architecture for deep convolutional neural network.

of which the stripe noise is future reduced and the solution is improved. The second layer groups together the low-noise and high-resolution feature maps to generate the final low-noise and high-resolution image.

The above two operations constitute a convolutional neural network. Fig. 2 illustrates this network in detail. Next we formulate each operation.

3.1.1 Image Denoising: As shown by the state-of-the-art methods, each prior information can be considered as a row convolution kernel. In our formulation, the first layer is consisted of a set of row convolution kernels that will be optimized into the network learning. Formally, the first layer is formulated as:

$$M_1(\boldsymbol{Y}) = \max(0, W_1 * \boldsymbol{Y} + B_1), \tag{1}$$

where W_1 stands for the filters, B_1 represents the biases and '$*$' is the convolution operation. Note that the size of each filter is $c \times 1 \times f_1$, where c represents the number of channels of the image, the number 1 denotes the height of the filter and f_1 is the width of the filter. In this paper we only use gray-scale images for training, so c is set to 1. We adopt n_1 filters to convolve the image, and get n_1 feature maps corresponding to the filters. B_1 is an n_1-dimensional vector, whose each bias is added to the corresponding feature map. We use an activation function called Rectified Linear Unit ($\max(0, x)$) [23] on the feature maps. We can add more row convolutional layers to weaken the stripe noise. But it will aggravate the detail lost and thus requires more square convolutional layers to restore details at the cost of training time.

3.1.2 Image Denoising and Super Resolution: The first layer generates a set of feature maps. In the second operation, we apply two convolutional layers to nonlinearly map each feature maps

into the final image. The two layers are expressed as follows:

$$M_2(\boldsymbol{Y}) = \max(0, W_2 * M_1(\boldsymbol{Y}) + B_2), \tag{2}$$

$$M(\boldsymbol{Y}) = W_3 * M_2(\boldsymbol{Y}) + B_3, \tag{3}$$

Here W_2 corresponds to n_2 filters of size $n_1 \times f_2 \times f_2$, and W_3 contains a filter of a size $n_2 \times f_3 \times f_3$. B_2 represents a n_2 -dimensional vector and B_3 is a c-dimensional vector. We use a set of square convolution kernels to further eliminate the residual stripe noise in the feature maps and restore the details, and generate n_2 low-noise and high-resolution feature maps. Finally, we use a filter of a size $n_2 \times f_3 \times f_3$ to produce the final image.

Interestingly, although stripe noise removal and super resolution belong to two different image processing, they can all be considered as a convolution operation. We combine them by a convolutional neural network (Fig. 2). In this network, all the parameters are optimized by training. We detail the training in the following section.

3.2 Training

This section we detail the optimization of the network parameters $\theta = \{W_1, W_2, W_3, B_1, B_2, B_3\}$. The parameters are updated by minimizing the loss between the original clean images $\boldsymbol{X}$ and the corresponding network output images $M(\boldsymbol{Y}; \theta)$. Given some sample images $\{\boldsymbol{Y}_i\}$ and label images $\{\boldsymbol{X}_i\}$, the loss function is formulated as

$$L(\theta) = \frac{1}{n} \sum_{i=1}^{n} \left\| M(\boldsymbol{Y}_i; \theta) - X_i \right\|^2, \tag{4}$$

where n stands for the number of training samples. The training results favor a high PSNR. PSNR is the most commonly used to measure the quality of reconstruction of the image, and is an approximation to human perception of reconstruction quality. Although a higher PSNR generally indicates that the reconstruction is of higher quality, in some cases it may not. We use other evaluation metrics, e.g., SSIM, NQM to measure the quality and also observe satisfactory results (see Section 4.3).

We use stochastic gradient descent method to minimize the loss function. Especially, the weight parameters are updated with standard backpropagation as

$$\Delta_{i+1} = 0.9 \cdot \Delta_i - \eta \cdot \frac{\partial L}{\partial W_i^l}, \tag{5}$$

$$W_{i+1}^l = W_i^l + \Delta_{i+1}, \eta_{i+N} = \eta_i \cdot (1 + g \cdot N)^{-p}, \tag{6}$$

where $l \in \{1, 2, 3\}$ represents the number of the layers and i stands for iterations. g and p are gamma and power parameters, η is the learning rate and N is the number of the updated interval iterations of the learning rate. In contrast to [13], our learning rate is decreasing with the increase of the iteration number. This is important for the model to converge. All other parameter settings remain the same as the parameter settings in [13].

Before training, given a set of original clean images, we first add the stripe noise with a Gaussian distribution to generate a set of corresponding noisy images. We randomly crop the clean and the corresponding noisy images to produce a set of small images. The size of the small clean images (label image) is $(f_{sub\_h} - f_2 - f_3 + 2) \times (f_{sub\_w} - f_1 - f_2 - f_3 + 3) \times b$, avoiding border effects, and the size of the small noisy images (input image) is $f_{sub\_h} \times f_{sub\_w} \times b$, where b is the batch size.

In the training phase, the small noisy images are used as the input images of the network and the network produces smaller output images, of which the size is the same as the size of the small label images. The loss function is calculated only by the difference between the pixels of the network output images and the label images. After training, the network can be used to remove the stripe noise in the images of arbitrary sizes. We train our model using the Caffe package [24]. The graphics card model is GTX 1080.

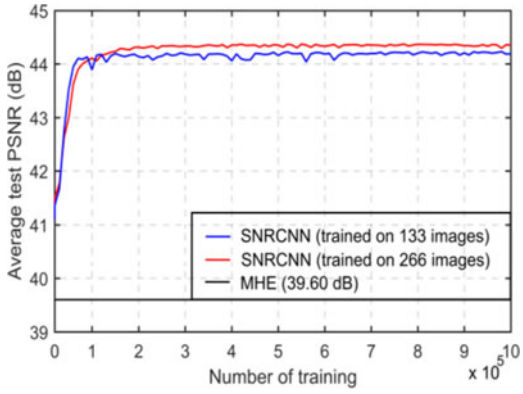


Fig. 3. Training with more training samples achieves a higher PSNR value.

4. Experiments

In this section, we examine the network performance to the network parameter settings. We first study the impact of the number of training samples on the network performance. Then we investigate the filter parameter settings like number of filters, size of the row convolution kernel, size of the square convolution kernel and number of layers. Finally, we compare the proposed approach with the state-of-the-art methods quantitatively and qualitatively.

4.1 Number of Training Samples

In this paper we use a small training set that contains 266 images downloaded from http://decsai.ugr.es/cvg/dbimagenes/. To compare the impact of the number of training samples on restoration quality, we randomly use half of 266 images as a smaller training set. The size of the input image in the network is $f_{sub\_h} = 59$ and $f_{sub\_w} = 67$. So we can get 62869 and 141132 training samples cropped from the 133-images and 266-images with a stride of 17, respectively. In order to distinguish between the row convolution kernel and the square convolution kernel, we add a box to the number of the row convolution kernel. The basic network parameters are set to $f_1 = \boxed{9}$, $f_2 = 5$, $f_3 = 5$, $n_1 = 64$ and $n_2 = 32$. The validation set contains 18 infrared images. The stripe noise we add on the training set and validation set is a Gaussian distribution of zero mean and standard deviation 0.02 and 0.01, respectively. We employ the MHE method as the baseline. The number of backpropagations is 10^{-6}.

Fig. 3 shows the average test PSNR curves using different numbers of training samples. More training samples mean that the network needs more training time to converge. So the average test PSNR of SNRCNN (trained on 133 images) is higher than that of SNRCNN (trained on 266 images) when the number of training is less than 1×10^5. We can see that with the increase in the number of the training samples, SNRCNN trained on 266 images obtains a PSNR of 44.35 dB, which is higher than 44.18 dB obtained by SNRCNN trained on 133 images. The results show that the restoration quality can be further improved with more training samples, but the high-level visual promotion is not so obvious. It is possible that the 133 images have already contained sufficient structure information. In addition, using a larger training set requires more computer memory, in particular using ILSVRC 2013 ImageNet that contains 395909 images. Therefore, we use 266 images as the default training set in the following experiments.

4.2 Model and Performance Trade-offs

Obviously, the more the number of the layers of row convolution kernels is, the lower the resolution of the image is. On the contrary, too many layers of square convolution kernel need to spend more

TABLE 1
Comparisons of Different Filter Numbers in Our SNRCNN

Mean, S.D	$n_1 = 32, n_2 = 16$	$n_1 = 64, n_2 = 32$	$n_1 = 128, n_2 = 64$
	Average test PSNR (dB)		
0, 0.005	45.7746	45.6048	**45.8803**
0, 0.01	44.3596	44.3486	**44.4121**
0, 0.015	42.7898	**42.9397**	42.8485
0, 0.02	40.4410	**40.5906**	40.4351

TABLE 2
Comparisons of Different Size of the Row Convolution Kernel

Mean, S.D	$f_1 = \boxed{11}$	$f_1 = \boxed{9}$	$f_1 = \boxed{7}$
	Average test PSNR (dB)		
0, 0.005	45.4855	**45.6233**	45.3886
0, 0.01	44.2626	**44.3537**	44.2359
0, 0.015	**42.6718**	42.6102	42.6314
0, 0.02	**40.9616**	40.7170	40.8006

training time. So in this section, we will progressively modify the model parameters to explore the trade-off between image quality and speed. The basic network parameters are $f_1 = \boxed{9}$, $f_2 = 5$, $f_3 = 5$, $n_1 = 64$ and $n_2 = 32$. The number of training samples is set to 120000 for all experiments.

4.2.1 Filter Number: Normally, increasing the number of filters in each layer can improve the restoration quality, at the cost of training time. We performed two experiments: (i) one adopts more filter numbers with $n_1 = 128$ and $n_2 = 64$, and (ii) the other uses fewer filter numbers with $n_1 = 32$ and $n_2 = 16$. We add the stripe noise of four different Gaussian distributions on the images in the validation set (see Table 1). All other parameter settings in the two experiments are consistent with Section 4.1. The results in Table 1 shows that it is not "the more the better". The network performance is also related to the amplitude and frequency of the stripe noise. It is mainly because that heavy stripe noise has great interference to image details, which are almost masked.

4.2.2 Size of the Row Convolution Kernel: In general, the stripe noise can be further suppressed if we increase the width of the row convolution kernel, yet the details will be lost more. Specifically, the filter sizes in the second and third layers are fixed to $f_2 = 5$ and $f_3 = 5$, then we increase and reduce the width of the filters in the first layer to (i) $f_1 = \boxed{11}$ ($\boxed{11} - 5 - 5$) and (ii) $f_1 = \boxed{7}$($\boxed{7} - 5 - 5$). Table 2 shows the results calculated at 10^{-6} backpropagations. We can observe that a larger width achieves higher PSNR values of 42.6718 dB and 40.9616 dB when the images have a serious stripe noise. On the contrary, when the images have a slight stripe noise, a moderate width of $f_1 = \boxed{9}$ achieves the highest PSNR value. This is mainly because it achieves a trade-off between

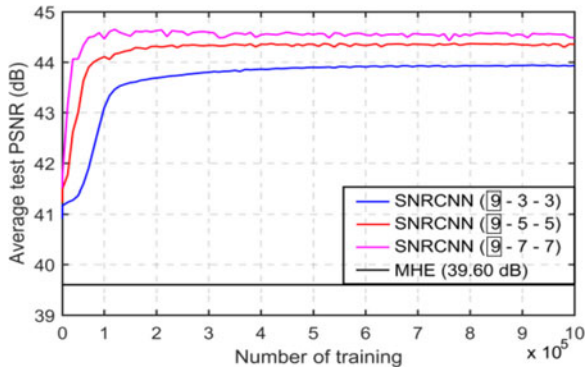


Fig. 4. Larger filter size in second and third layers promotes network performance.

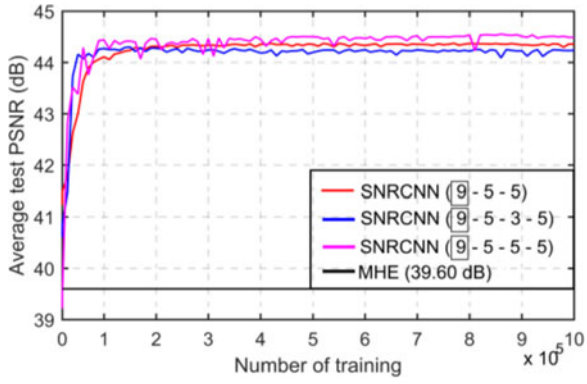


Fig. 5. More layers do not always achieve better network performance.

stripe noise suppression and detail losing. In most cases the stripe noise is slight, so the filter width in the first layer is set to $\boxed{9}$.

4.2.3 Size of the Square Convolution Kernel: In this section, we examine the restoration quality to the size of the square convolution kernel. Specifically, we fix the width of the row convolution kernel in the first layer to be $f_1 = \boxed{9}$, while expanding the filter size of other layers to $f_2 = f_3 = 7$ and narrowing the filter size of other layers to $f_2 = f_3 = 3$. Convergence curves in Fig. 4 show that enlarge the filter size of square convolution kernel could grasp richer image detail to improve the network performance.

However, using a larger filter size will increase the running time. For instance, $\boxed{9} - 3 - 3$, $\boxed{9} - 5 - 5$ and $\boxed{9} - 7 - 7$ have 19296, 52576 and 102496 parameters, respectively. The number of parameters of $\boxed{9} - 7 - 7$ is almost twice that of $\boxed{9} - 5 - 5$ and five times of $\boxed{9} - 3 - 3$. By weighing network performance and running time, the filter sizes in second and third layers are set to 5.

4.4.4 Number of Layers: Research in [29] shows that adding more layers moderately could improve CNN. Obviously, increasing the layer of the row convolution kernel means that a deeper structure is needed to deal with the problem of losing blur. Here, we only add a layer of square convolution kernel between the second layer and the third layer, which has 32 filters. We conduct two experiments, i.e., $\boxed{9} - 5 - 3 - 5$ and $\boxed{9} - 5 - 5 - 5$. The initial parameters of the additional layer remain the same with the second layer.

Convergence curves in Fig. 5 show that not all deeper network architectures lead to better network performance, which is also related to the filter size of additional layer. Although the $\boxed{9} - 5 - 5 - 5$

TABLE 3
The Average Results of Using State-of-the-Art Methods on Set5

Eval. Mat	Mean, S.D	TV	GF	NLM	BM3D	MHE	SNRCNN
PSNR	0, 0.01	26.0632	29.8759	33.2507	31.6000	33.5965	**40.8761**
	0, 0.015	25.9710	29.7880	33.0673	31.3442	33.3895	**39.6909**
	0, 0.02	25.8278	29.6989	32.7520	30.8426	33.1585	**38.2025**
SSIM	0, 0.01	0.7877	0.9373	0.8904	0.8744	0.9543	**0.9846**
	0, 0.015	0.7779	0.9364	0.8901	0.8661	0.9534	**0.9823**
	0.02	0.7604	0.9344	0.8877	0.8411	0.9508	**0.9778**
NQM	0, 0.01	21.1766	24.9893	28.3641	26.7134	28.7099	**36.0770**
	0, 0.015	21.0844	24.9015	28.1807	26.4576	28.5029	**34.8917**
	0, 0.02	20.9412	24.8124	27.8654	25.9561	28.2719	**33.4033**

network shows better results than that of $\boxed{9}-5-5$ network, it increases both the complexity of the model and the running time. In addition, when we try a deeper network (5 layers), we find that it is difficult to initialize the appropriate parameters to ensure convergence as shown in [30].

4.3 Model and Performance Trade-Offs

Here, we show the quantitative and qualitative evaluation of our method against state-of-the-art approaches. The network we choose contains three layers with parameters $f_1 = \boxed{9}$, $f_2 = 5$, $f_3 = 5$, $n_1 = 64$ and $n_2 = 32$, and the training set contains 266 images. We add the stripe noise from Gaussian distributions with zero mean and standard deviation of 0.02 on the training set and add the stripe noise with three different standard deviations of 0.01, 0.015 and 0.02 on the test set. The number of backpropagations is 10^{-6}.

Comparisons: We compare our method with the state-of-the-art stripe noise removal approaches:

1) TV-total variation method [11]
2) GF-guided filter method [13]
3) NLM-Non-Local Means (NLM) [14]
4) BM3D- block method of 3-dimension [15]
5) MHE-midway histogram equalization [10]

Test set: Color images in Set 5 [25] and Set 14 [26] contains more image detail than infrared images. Thus, if a good result is obtained on color images, better results are naturally obtained on the infrared images as well. Of course, we also use infrared image set that contains 18 images to evaluation the network performance.

Evaluation metrics: PSNR, structural similarity index (SSIM) [26] and noise quality measure (NQM) [28] are used to evaluate restoration quality.

4.3.1 Quantitative and Qualitative Evaluation: Results in Tables 3–5 indicate that our SNRCNN achieves the highest scores and significantly outperforms other state-of-the-art approaches in all evaluation matrices. Meanwhile, Figs. 6–8 show the comparisons of different methods. We can clearly find that the results using TV method have the most serious detail lost and the lowest score. The results using GF method, NLM method and BM3D almost remove the stripe noise but loses detail, and the results using MHE method produce artifacts across the image. In contrast, our SNRCNN achieves the best reconstruction quality without artifacts and detail lost.

TABLE 4
The Average Results of Using State-of-the-Art Methods on Set14

Eval. Mat	Mean, S.D	TV	GF	NLM	BM3D	MHE	SNRCNN
PSN	0, 0.01	24.9671	30.0454	31.5457	30.2040	33.7061	**40.5710**
	0, 0.015	24.9086	29.9988	31.3992	30.0326	33.5582	**39.2040**
	0, 0.02	24.8147	29.9165	31.1819	29.7090	33.3951	**37.6877**
SSIM	0, 0.01	0.6841	0.9428	0.8464	0.8172	0.9655	**0.9854**
	0, 0.015	0.6767	0.9420	0.8458	0.8102	0.9647	**0.9829**
	0.02	0.6668	0.9406	0.8441	0.7942	0.9641	**0.9779**
NQM	0, 0.01	20.1345	25.2128	26.7131	25.3714	28.8736	**35.7749**
	0, 0.015	20.0760	25.1663	26.5666	25.2037	28.7257	**34.4079**
	0, 0.02	19.9821	25.0839	26.3493	24.8764	28.5626	**32.8916**

TABLE 5
The Average Results of Using State-of-the-Art Methods on Set of 18 Infrared Images

Eval. Mat	Mean, S.D	TV	GF	NLM	BM3D	MHE	SNRCNN
PSNR	0, 0.01	35.1503	38.7946	36.9817	38.2820	39.6122	**44.4024**
	0, 0.015	34.1913	38.2235	36.8851	36.6032	39.0166	**42.6430**
	0, 0.02	33.0919	37.7479	36.6646	34.5310	38.5393	**40.6585**
SSIM	0, 0.01	0.9052	0.9730	0.9249	0.9313	0.9727	**0.9876**
	0, 0.015	0.8800	0.9718	0.9248	0.9022	0.9716	**0.9858**
	0.02	0.8413	0.9704	0.9232	0.8393	0.9703	**0.9791**
NQM	0, 0.01	28.1176	31.7619	29.9490	31.2493	32.5795	**37.4566**
	0, 0.015	27.1586	31.1908	29.8524	29.5705	31.9839	**35.7899**
	0, 0.02	26.0592	30.7152	29.6320	27.4983	31.5066	**34.0170**

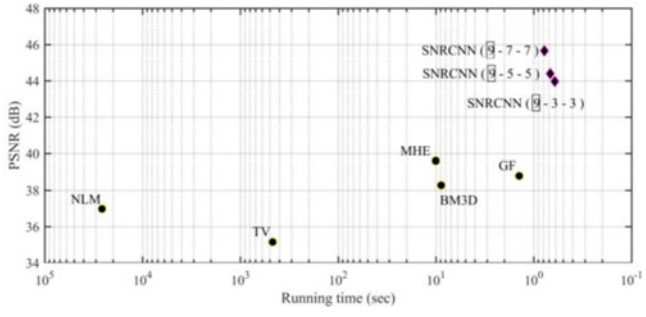


Fig. 6. The "bird" image from Set 5 with the stripe noise of the standard deviation of 0.015.

Fig. 7. The “comic” image from Set14 with the stripe noise of the standard deviation of 0.02.

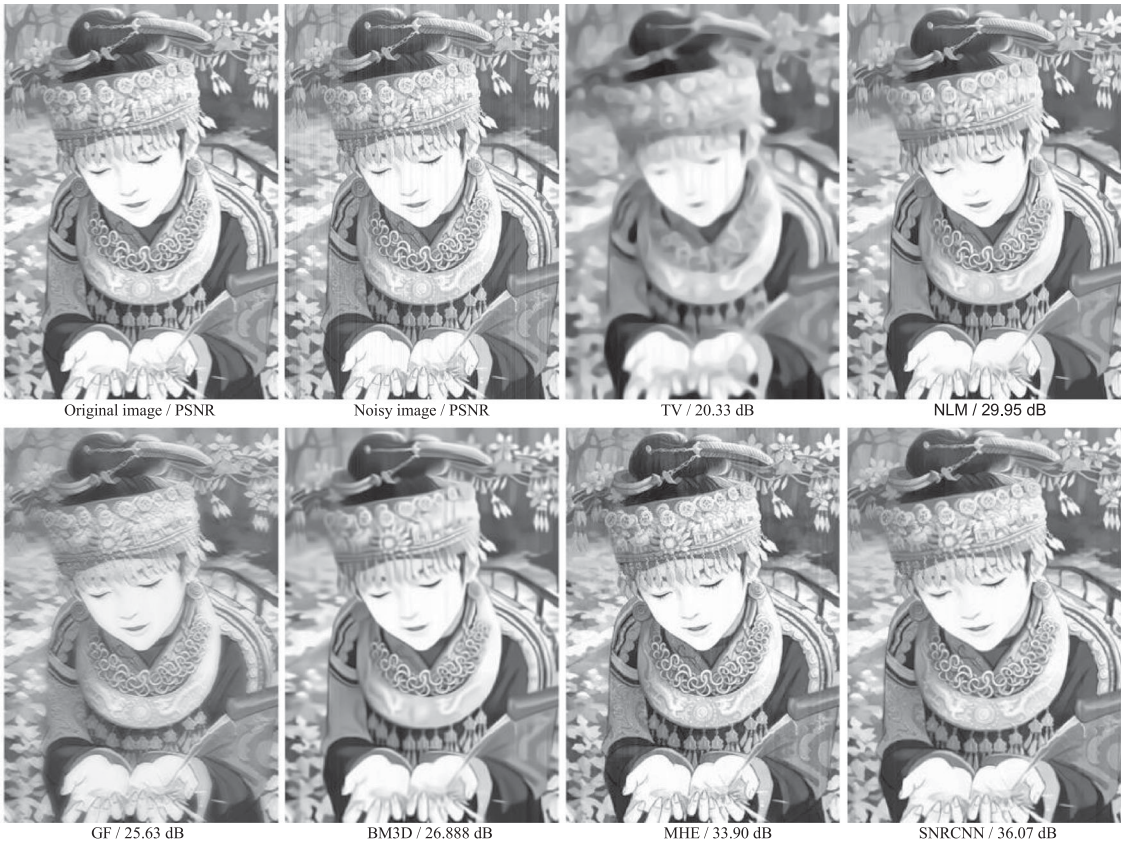


Fig. 8. The “street” image from the infrared image set with the stripe noise of the standard deviation of 0.005.

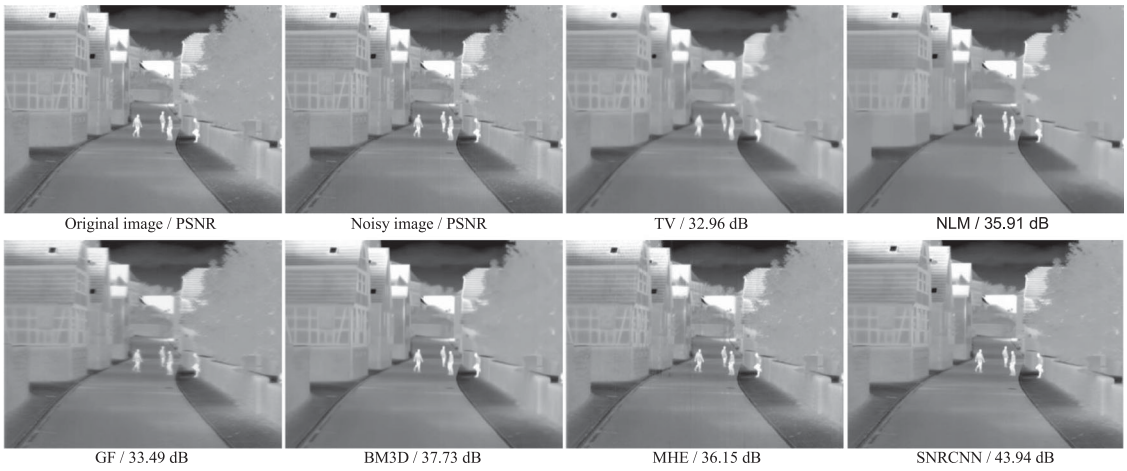


Fig. 9. Our SRNCNN achieves the highest PSNR value, whilst gets the fastest speed.

4.3.2 Running Time: Fig. 9 is the running time results of different approaches on the infrared image set with the stripe noise of the standard deviation of 0.01. All the methods are implemented by MATLAB. The results are obtained using the same computer (Intel CPU 3.4 GHz, 16 GB memory and GTX 1080). Note that we run our SNRCNN algorithm on the GPU and obtain the running time result. We can observe that the $\boxed{9}-7-7$ network yields the highest score while the $\boxed{9}-3-3$ network is the fastest. Our SNRCNN is superior to other state-of-the-art methods in performance and speed. Although our three different networks have similar running time, their training time is significantly different. Therefore, we need to consider the trade-off between training time and performance in practice.

5. Conclusion

We have proposed a novel infrared image destriping method called SNRCNN. Our main contributions are to combine super-resolution and image denoising through deep learning network. Compared with existing state-of-the-art methods, our SNRCNN can generate more accurate and reliable destriping results while better preserving the detail of the infrared images. High-quality destriping results bear out the effectiveness of this method.

References

[1] J. M. Mooney and F. D. Shepherd, "Characterizing IR FPA nonuniformity and IR camera spatial noise," *Infrared Phys. Technol.*, vol. 37, no. 5, pp. 595–606, 1996.
[2] J. M. Mooney, F. D. Shepherd, W. S. Ewing, J. E. Murguia, and J. Silverman, "Responsivity nonuniformity limited performance of infrared staring cameras," *Opt. Eng.*, vol. 28, pp. 1151–1161, 1989.
[3] M. Schulz and L. Caldwell, "Nonuniformity correction and correctability of infrared focal plane arrays," *Proc. SPIE*, vol. 2470, pp. 200–211, 1995.
[4] W. Zhao and C. Zhang, "Scene-based nonuniformity correction and enhancement: Pixel statistics and subpixel motion," *J. Opt. Soc. Amer. A*, vol. 25, pp. 1668–1681, 2008.
[5] J. G. Harris and Y. M. Chiang, "Nonuniformity correction of infrared image sequences using the constant-statistics constraint," *IEEE Trans. Image Process.*, vol. 8, no. 8, pp. 1148–1151, Aug. 1999.
[6] D. A. Scribner, K. A. Sarkady, M. R. Kruer, and J. T. Caulfield, "Adaptive nonuniformity correction for IR focal plane arrays using neural networks," *Proc. SPIE*, vol. 1541, pp. 100–109, 1999.
[7] B. M. Ratliff and M. M. Hayat, "An algebraic algorithm for nonuniformity correction in focal-plane arrays," *J. Opt. Soc. Amer. A*, vol. 19, pp. 1737–1747, 2002.
[8] B. Narayanan, R. C. Hardie, and R. A. Muse, "Scene-based nonuniformity correction technique that exploits knowledge of the focal-plane array readout architecture," *Appl. Opt.*, vol. 44, pp. 3482–3491, 2005.
[9] J. E. Pezoa and M. M. Hayat, "Multimodel Kalman filtering for adaptive nonuniformity correction in infrared sensors," *J. Opt. Soc. Amer. A*, vol. 23, pp. 1282–1291, 2006.

[10] Y. Tendero, J. Gilles, S. Landeau, and J. Morel, "Efficient single image non-uniformity correction algorithm," *Proc. SPIE,* vol. 7834, pp. 10–22, 2010.
[11] E. Vera, P. Meza, and S. Torres, "Total variation adaptive scene-based nonuniformity correction," in *Proc. Imag. Syst.*, 2010, Paper JTuA24.
[12] C. Dong, C. C. Loy, K. He, and X. Tang, "Image super-resolution using deep convolutional networks," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 38, no. 2, pp. 295–307, Feb. 2016.
[13] Y. P. Caol, M. Y. Yang, and C.-L. Tisse, "Effective strip noise removal for low-textured infrared images based on 1D guided filtering," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 26, no. 12, pp. 1051–8215, Dec. 2015.
[14] A. Buades, B. Coll, and J. M. Morel, "Nonlocal image and movie denoising," *Int. J. Comput. Vis.*, vol. 76, no. 2, pp. 123–139, 2008.
[15] K. Dabov, A Foi, V Katkovnik, and K. Egiazarian, "Image denoising by sparse 3D transform-domain collaborative filtering," *IEEE Trans. Image Process.*, vol. 16, no. 8, pp. 2080–2095, Aug. 2007.
[16] X. B. Sui, Q. Chen, and G. H. Gu, "Algorithm for eliminating stripe noise in infrared image," *J. Infrared Millim. Waves*, vol. 31, no. 2, pp. 106–112, 2012.
[17] K. He, X. Zhang, S. Ren, and J. Sun, "Spatial pyramid pooling in deep convolutional networks for visual recognition," in *Proc. Eur. Conf. Comput. Vis.*, 2014, pp. 346–361.
[18] A. Krizhevsky, I. Sutskever, and G. Hinton, "Imagenet classification with deep convolutional neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2012, pp. 1097–1105.
[19] W. Ouyang *et al.*, "Deepid-Net: Multi-Stage and deformable deep convolutional neural networks for object detection," arXiv preprint, arXiv:1409.3505, 2014.
[20] C. Szegedy, S. Reed, D. Erhan, and D. Anguelov, "Scalable, highquality object detection," arXiv preprint, arXiv:1412.1441, 2014.
[21] N. Zhang, J. Donahue, R. Girshick, and T. Darrell, "Part-based RCNNs for fine-grained category detection," in *Proc. Eur. Conf. Comput. Vis.*, 2014, pp. 834–849.
[22] Y. Sun, Y. Chen, X. Wang, and X. Tang, "Deep learning face representation by joint identification-verification," in *Proc. Adv. Neural Inf. Process. Syst.*, 2014, pp. 1988–1996.
[23] V. Nair and G. E. Hinton, "Rectified linear units improve restricted Boltzmann machines," in *Proc. Int. Conf. Mach. Learn.*, 2010, pp. 807–814.
[24] Y. Jia *et al.*, "Caffe: Convolutional architecture for fast feature embedding," in *Proc. 22nd Int. Conf. ACM Multimedia*, 2014, pp. 675–678.
[25] M. Bevilacqua, A. Roumy, C. Guillemot, and M. L. A. Morel, "Low complexity single-image super-resolution based on nonnegative neighbor embedding," in *Proc. Brit. Mach. Vis. Conf.*, 2012, pp. 1–10.
[26] R. Zeyde, M. Elad, and M. Protter, "On single image scale-up using sparse-representations," in *Proc. Curves Surf.*, 2012, pp. 711–730.
[27] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: From error visibility to structural similarity," *IEEE Trans. Image Process.*, vol. 13, no. 4, pp. 600–612, Apr. 2004.
[28] N. Damera-Venkata, T. D. Kite, W. S. Geisler, B. L. Evans, and A. C. Bovik, "Image quality assessment based on a degradation model," *IEEE Trans. Image Process.*, vol. 9, no. 4, pp. 636–650, Apr. 2000.
[29] K. He and J. Sun, "Convolutional neural networks at constrained time cost," arXiv preprint, arXiv:1412.1710, 2014.
[30] D. Glasner, S. Bagon, and M. Irani, "Super-resolution from a single image," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2009, pp. 349–356.