Rain Streaks Removal for Single Image via Kernel-Guided Convolutional Neural Network

Ye-Tao Wang, Xi-Le Zhao, *Member, IEEE,* Tai-Xiang Jiang, *Member, IEEE*, Liang-Jian Deng, *Member, IEEE*, Yi Chang, *Member, IEEE*, and Ting-Zhu Huang

***Abstract*— Recently emerged deep learning methods have achieved great success in single image rain streaks removal. However, existing methods ignore an essential factor in the rain streaks generation mechanism, i.e., the motion blur leading to the line pattern appearances. Thus, they generally produce overderaining or underderaining results. In this article, inspired by the generation mechanism, we propose a novel rain streaks removal framework using a kernel-guided convolutional neural network (KGCNN), achieving state-of-the-art performance with a simple network architecture. More precisely, our framework consists of three steps. First, we learn the motion blur kernel by a plain neural network, termed parameter network, from the detail layer of a rainy patch. Then, we stretch the learned motion blur kernel into a degradation map with the same spatial size as the rainy patch. Finally, we use the stretched degradation map together with the detail patches to train a deraining network with a typical ResNet architecture, which produces the rain streaks with the guidance of the learned motion blur kernel. Experiments conducted on extensive synthetic and real data demonstrate the effectiveness of the proposed KGCNN, in terms of rain streaks removal and image detail preservation.**

***Index Terms*— Convolutional neural network (CNN), motion blur kernel, rain streaks generation mechanism, rain streaks removal.**

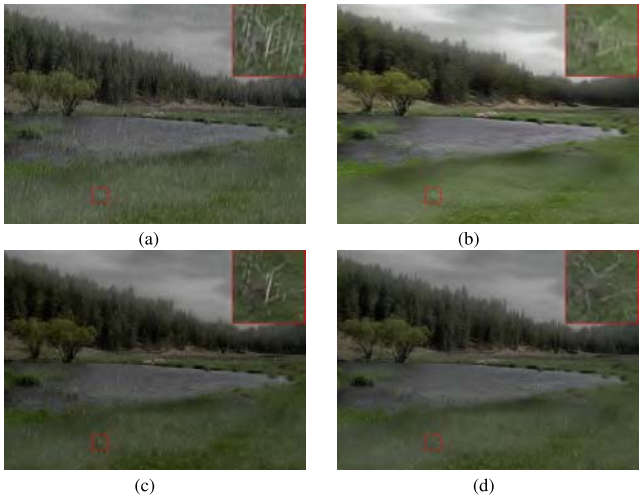


Fig. 1. Example of rain streaks removal for a real-world rainy image. (a) Rainy image. (b)–(d) Deraining results by DID [11], DDN [12], and the proposed KGCNN, respectively.

I. Introduction

OUTDOOR vision systems are frequently affected by bad weather conditions, one of which is the rain. Because of the high motion velocities and the light scattering, raindrops usually introduce bright streaks into the images or videos acquired by cameras. This undesirable interference will degrade the performance of various computer vision algorithms [1], such as object detection [2], [3], event detection [4], action recognition [5], [6], and scene analysis [7], [8]. Rain streaks removal aims at alleviating the effects of the rain, which is an essential task [9] and has been investigated extensively [10]. Fig. 1 shows one example of the single image rain streaks removal.

The central issue in deraining methods is to exploit the discriminative characteristics of the rain streaks and the clean image. Model-based methods formulate rain streaks removal as an optimization problem involved handcrafted regularizers expressing prior knowledge of the solution, such as the high-frequency (HF) property [13], [14], directionality [15]–[18], and repeatability [19] of the rain streaks and the piecewise smoothness [18]–[21] of the image. However, these model-based methods generally cannot deal with real rainy scenarios since the degradation processes can be very complex. To overcome this shortage, learning-based methods attempt to learn the discriminative characteristics from the data, such as learned dictionaries [22], the Gaussian mixture models (GMMs) [20], [21], the stochastic distributions [23], and the convolutional filters [24]–[26]. Recently emerged deep learning methods [11], [12], [27]–[39] capture the data knowledge by a trained deep neural network with

Manuscript received October 6, 2018; revised June 3, 2019 and June 4, 2020; accepted August 8, 2020. This work was supported in part by NSFC under Grant 61876203, Grant 61772003, and Grant 61702083 and in part by the Fundamental Research Funds for the Central Universities under Grant JBK2001011 and Grant JBK2001035. *(Corresponding author: Xi-Le Zhao.)*

Ye-Tao Wang, Xi-Le Zhao, Liang-Jian Deng, and Ting-Zhu Huang are with the School of Mathematical Sciences, University of Electronic Science and Technology of China, Chengdu 611731, China (e-mail: munaiyi719@gmail.com; xlzhao122003@163.com; tingzhuhuang@126.com; liangjian1987112@126.com).

Tai-Xiang Jiang is with the FinTech Innovation Center, School of Economic Information Engineering, Southwestern University of Finance and Economics, Chengdu 611130, China (e-mail: taixiangjiang@gmail.com; jiangtx@swufe.edu.cn).

Yi Chang is with the National Key Laboratory of Science and Technology on Multispectral Information Processing, School of Artificial Intelligence and Automation, Huazhong University of Science and Technology, Wuhan 430074, China, and also with the AI Center, Peng Cheng Laboratory, Shenzhen 518000, China (e-mail: yichang@hust.edu.cn).

This article has supplementary downloadable material available at http://ieeexplore.ieee.org, provided by the authors.

Color versions of one or more of the figures in this article are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TNNLS.2020.3015897

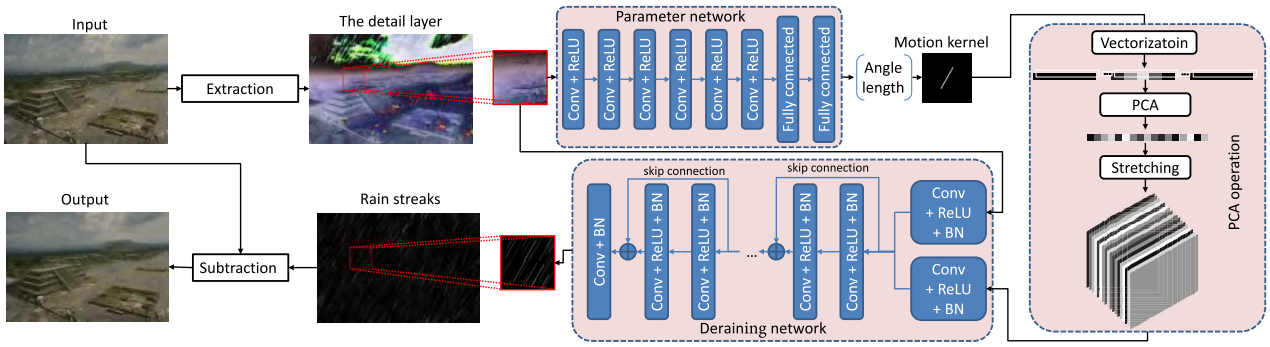


Fig. 2. Flowchart of the proposed single image rain streaks removal framework. The rainy image is decomposed into the detail layer and the base layer. A detail patch is fed to the parameter network to obtain the angle and the length of the motion blur kernel. The motion blur kernel is stretched to the degradation map. The detail patch and the degradation map are then fed into the deraining network, whose output is the rain streaks patch. Finally, the deraining image is obtained by the subtraction of the rainy image and the rain streaks image.

strong representation ability, leveraging the data to the most extent and obtaining promising results.

Despite achieving great success in rain streaks removal, existing deep learning methods still face two challenges. First, the performance of the trained network highly relies on the training data. Examples include the deep detail network (DDN) [12], [40], which concentrated on learning the nonlinear mapping from the detail layer to the rain streaks with straightforward network architectures. Second, many deep learning methods suffer from the burden of network designing. Examples include CNN-based methods, such as DID [11] and JORDER [29], which elaborately design the CNN architectures in order to tackle the complex situations of real rainy scenarios. Due to the abovementioned limitations, state-of-the-art methods cannot effectively distinguish the rain streaks and the line pattern textures, leading to either underderaining results (remaining rain streaks) or overderaining results (erasing image details). Fig. 1 shows a typical example, in the enlarged areas of which we can observe that DID erased some grasses (overderaining) and DDN retains obvious rain streaks (underderaining).

To tackle the abovementioned challenges, we propose a novel deep learning method for rain streaks removal. The basic idea is to guide rain streaks removal via the rain streaks generation mechanism, which is modeled as a convolution of a raindrop mask with a motion blur kernel. As pointed out in [1], the appearance of the rain streaks is mainly related to the motion blur. This mechanism naturally exploits two important distinguishing characteristics of the rain streaks, i.e., the repeatability and the directionality. Specifically, the repeatability of the rain streaks, which has also been mentioned in [19], can be utilized to estimate the motion blur kernel. Then, the kernel, which contains the information of the rain streaks' direction, is supposed to guide the deraining stage. Therefore, this would contribute to distinguishing the rain streaks and the line pattern textures in different directions. Once established, two questions arise: 1) how to infer the motion kernel from the data? and 2) how to utilize the information provided by the motion blur kernel in the process of removing rain streaks?

The answers to the abovementioned questions lead to a three-step framework for rain streaks removal (see Fig. 2 for the flowchart). In our approach, we assume that the rain streaks in a small patch approximately share the same motion blur kernel (i.e. direction). First, extract the detail patch from a rainy patch. Then, feed the rainy patch to the parameter network, a plain network with six "Conv + ReLU" layers and two fully connected layers to infer the angle and the length of the motion blur kernel. To enable the learned motion blur kernels to participate in the subsequent rain streaks removal process, we adopt the dimensionality stretching strategy [41], which stretched the motion blur kernels to degradation maps with the same spatial size as the detail patches. Then, the detail patch together with the degradation map is inputed to a 26-layer network of the typical ResNet architecture, and its output is the patch of rain streaks. Finally, we obtain the rain streaks removal results by subtracting the rain streaks image from the rainy image. The core idea of our framework is exploiting the generation mechanism of the rain streaks to guide rain streaks removal, and the flowchart of our framework is shown in Fig. 2. In this work, we mainly focus on the rain streaks removal and the readers may apply haze removal methods [42], [43] as a preprocessing or postprocessing step for heavy rain accompanied by haze.

The contributions of this article mainly include three aspects.

1) We build a novel rain streak observation model based on the motion blur mechanism, which enables us to utilize the repeatability and the directionality of the rain streaks. To infer the parameters (length and angle) of the motion blur kernel, we design a subnetwork, i.e., the parameter network, by exploiting the rain streaks generation mechanism.
2) We propose an effective kernel-guided CNN (KGCNN) with a simple architecture for rain streaks removal, in which the estimation process is thoroughly guided by the automatically learned motion blur kernel.
3) Extensive experiments are conducted on publicly available real and synthetic data. Qualitative and quantitative comparisons with existing state-of-the-art methods

suggest that the proposed KGCNN achieves state-of-the-art performance in terms of removing rain streaks and keeping textures.

The organization of this article is as follows. We provide an overview of the existing deraining methods in Section II. Section III gives the detailed architecture of the proposed KGCNN. In Section IV, experimental results on synthetic data and real-world data are reported. Finally, we draw conclusions in Section V.

II. Related Works

In the past decades, numerous methods have been proposed to improve the visibility of images/videos captured with rain streaks interference. Traditionally, these methods can be divided into two categories: single image deraining methods and multiple-image/video deraining methods. Nevertheless, the explosive development of deep learning brings in a novel branch, i.e., the deep learning methods.

A. Traditional Methods

1) Single Image Deraining Methods: For the single image deraining task, Kang *et al.* [13] decomposed a rainy image into low-frequency (LF) and HF components using a bilateral filter and then performed morphological component analysis (MCA)-based dictionary learning and sparse coding to separate the rain streaks in the HF component. To alleviate the loss of the details when learning HF image bases, Sun *et al.* [14] tactfully exploited the structural similarity of the derived HF image bases. Kim *et al.* [44] took advantage of the nonlocal similarity. Chen and Hsu [19] considered the similar and repeated patterns of the rain streaks and the smoothness of the rain-free content. Luo *et al.* [22] adopted discriminative sparse codes of the rain layer and the derained image layer. The recent work by Li *et al.* [21] utilized the GMM patch priors for rain streaks removal, with the ability to account for rain streaks of different orientations and scales. Meanwhile, the directional property of rain streaks received a lot of attention in [15]–[17] and [45] and these methods achieved promising performances. Ren *et al.* [46] removed the rain streaks from the image recovery perspective. Wang *et al.* [47] took advantage of image decomposition and dictionary learning.

2) Video Deraining Methods: Garg and Nayar [48] first raised a video rain streaks removal method with a comprehensive analysis of the visual effects of rain streaks on an imaging system. Since then, many approaches have been proposed for the video rain streaks removal task and obtained good performances with different rain circumstances. Tripathi and Mukhopadhyay [49] took the spatiotemporal properties into consideration. In [19], the similarity and repeatability of rain streaks were considered, and a generalized low-rank appearance model was proposed. Chen and Chau [50] considered highly dynamic scenes. Whereafter, Kim *et al.* [51] considered the temporal correlation of rain streaks and the low-rank nature of clean videos. Santhaseelan and Asari [52] detected and removed the rain streaks based on phase congruency features. In addition, comprehensive early existing video deraining methods were reviewed in [53]. You *et al.* [54] took the raindrops adhered to a windscreen or window glass into account. In [18], a novel tensor-based video rain streaks removal approach was proposed by considering the directional property. The rain streaks and the clean image were stochastically modeled as a mixture of Gaussians by Wei *et al.* [23]. The convolutional sparse coding (CSC), which has shown its ability in image cartoon-texture decomposition [26], was also adopted by Li *et al.* [24] for the video rain streaks removal. Ren *et al.* [55] addressed the video desnow and deraining task based on matrix decomposition.

B. Deep Learning-Based Methods

Deep learning was first applied to deraining in [56], in which a three-layer CNN was designed to remove static raindrops and dirt spots from pictures taken through window glass. Fu *et al.* [40] were the first to successfully tailor a CNN for the rain streaks removal task. Moreover, Fu *et al.* [12] designed the DDN to further improve the performance by adopting the well-known ResNet [57] structure. Pan *et al.* [58] simultaneously operated on the texture component and the structural component. Yang *et al.* [29] added a binary map that provides rain streak locations to the degradation model and proposed a deep recurrent network for simultaneous rain streaks detection and removal.

Chen *et al.* [28] proposed a CNN framework for video rain streaks removal, whereas Liu *et al.* [30], [31] adopted the recurrent neural network (RNN) for this task. For jointly rain-density estimation and deraining, Zhang and Patel [11] raised a density-aware multistream densely connected CNN (DID). Recently, the increasingly popular generative adversarial networks (GANs) were applied to deal with adherent raindrop [27]. Since the rainy scenarios can be very complex, the latest methods [32]–[34], [59]–[62] removed the rain streaks removal in a recursive/progressive/hierarchical manner.

III. Framework of the Kernel-Guided Convolutional Neural Network

This section describes the details of the framework of the proposed KGCNN. Section III-A gives our rain streaks degradation model. Sections III-B–III-D give three main parts, i.e., the parameter network, the dimensionality stretching, and the deraining network of the proposed rain streaks removal framework, respectively.

A. Degradation Model

Taking image patches as the basic unit, within which the rain streaks can be approximately viewed as sharing the same direction, we consider the basic additive degradation model [20], [21]: $\mathbf{O} = \mathbf{B} + \mathbf{R}$, where $\mathbf{O}$, $\mathbf{B}$, and $\mathbf{R} \in \mathbb{R}^{m\times n\times 3}$ are patches of the rainy image, the clean image, and the rain streaks, respectively. Because of the high velocity of the raindrops, we model the generation of rain streaks as the following motion blur process:

$$\mathbf{O} = \mathbf{B} + \mathbf{K}(\theta, l) \otimes \mathbf{M} \tag{1}$$

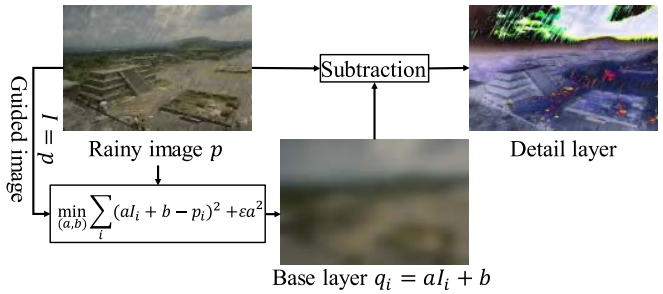


Fig. 3. Illustration of the extraction of the detail layer from the rainy image via the guided image filtering [63].

where θ and l are, respectively, the angle and length of the motion blur kernel $\mathbf{K}(\theta, l)$, $\mathbf{M} \in \mathbb{R}^{m\times n\times 3}$ is the raindrops mask, and $\otimes$ denotes the spatial convolution operator. The two important factors, i.e., the length l and the angle θ, essentially encode the motion blur kernel and can be easily inferred from the rainy images by exploiting the repeatability of the rain streaks.

Meanwhile, many existing methods assume that rain streaks mainly exist on the detail layer [12], [40], [58] or the HF layer [13], [14]. Following this research line, we adopt the guided image filtering [63] as the low-pass filter because it is simple and fast to implement.[1] Fig. 3 shows the decomposition of the rainy image into the base layer $\mathbf{O}_\mathrm{S}$ and the detail layer $\mathbf{O}_\mathrm{T}$ (denoted as "detail component" in [12], [40], and [58]), which satisfies $\mathbf{O} = \mathbf{O}_\mathrm{S} + \mathbf{O}_\mathrm{T}$. Assuming that the detail layer contains all rain streaks, the degradation model (1) turns to be

$$\mathbf{O}_\mathrm{T} = \mathbf{B}_\mathrm{T} + \mathbf{K}(\theta, l) \otimes \mathbf{M} \tag{2}$$

where $\mathbf{B}_\mathrm{T} \in \mathbb{R}^{m\times n\times 3}$ is the rain-free content of the detail layer and the goal is also transferred to estimate the clean detail part and separate the rain streaks from the rainy detail layer.

The advantages of processing on the detail layer have been fully discussed in [12] and [40]. In order to facilitate the readers, we briefly bring them here. It can be found in Fig. 2 that the detail layer contains all rain streaks, i.e., $\mathbf{O}_\mathrm{T} = \mathbf{B}_\mathrm{T} + \mathbf{R}$. Thus, training and testing on the detail layer $\mathbf{O}_\mathrm{T}$ is sufficient and compact. Meanwhile, compared with the original image, the detail layer is more sparse, which helps the network to focus on important information. In this work, considering the benefits of processing on the detail layer, we attempt to design and train a CNN derainer $\mathcal{F}_\mathrm{D}(\cdot; \Theta_\mathrm{D})$, which maps the detail patch $\mathbf{O}_\mathrm{T}$ into the rain streaks patch $\mathbf{R} = \mathbf{K}(\theta, l) \otimes \mathbf{M}$.

B. Parameter Network

We select the CNN to infer the motion blur kernel from an input detail patch, considering its overwhelming superiority on features extraction. As we pointed out above, the motion blur kernel within the generation of rain streaks is conclusively decided by two parameters, i.e., the angle θ and length l. We built the parameter network to learn the parameters[2] (length and angle) of the motion blur kernel. We adopt the CNN $\mathcal{F}_\mathrm{P}(\cdot; \Theta_\mathrm{P}) : \mathbb{R}^{m\times n\times 3} \to \mathbb{R}^2$, which maps the input detail patch $\mathbf{O}_\mathrm{T}$ to the parameter vector $[\theta, l]^T$, with the network parameter Θ_P. The loss function thereof for training is

$$L_\mathrm{P}(\Theta_\mathrm{P}) = \frac{1}{n}\sum_{i=1}^{n} \left\| \mathcal{F}_\mathrm{P}\left(\mathbf{O}_\mathrm{T}^i; \Theta_\mathrm{P}\right) - \mathbf{p}^i \right\|_F^2 \tag{3}$$

where $\mathbf{p} = [\theta, l]^T$ is the parameter vector, $\|\cdot\|_F$ denotes the Frobenius norm, and i indexes the patches and motion blur kernels. The architecture of $\mathcal{F}_\mathrm{P}(\cdot)$ (denoted as "parameter network") is shown in Fig. 2. Once the parameters θ and l are determined, the motion blur kernel $\mathbf{K}(\theta, l)$ can be uniquely constructed.

C. Dimensionality Stretching

After obtaining the motion blur kernel $\mathbf{K}(\theta, l)$, the question comes to how to "tell" our CNN derainer $\mathcal{F}_\mathrm{D}(\cdot; \Theta_\mathrm{D})$ the information of the motion blur kernel for a "precise strike" on the rain streaks. The input of our deraining network $\mathcal{F}_\mathrm{D}(\cdot; \Theta_\mathrm{D})$, which would be detailed in Section III-D, is supposed to be the detail patch together with the motion blur kernel learned from this detail patch, since the motion blur kernel consists of the prior knowledge of the rain streaks. By enforcing a local connectivity pattern between neurons of adjacent layers, CNNs exploit spatial locality and thus cannot extract the complete information of the motion blur kernel when we just concatenate the detail patch and motion blur. Hence, inspired by the CNN-based single-image super-resolution work [41], we adopt the dimensionality stretching strategy.

The dimensionality stretching strategy is schematically shown in Fig. 2. First, the motion blur kernel $\mathbf{K} \in \mathbb{R}^{p\times p}$ is vectorized into a vector $\mathbf{k} \in \mathbb{R}^{p^2}$. After the vectorization, $\mathbf{k}$ is projected onto a t-dimensional linear space by the principal component analysis (PCA) technique. Then, the projected vector $\mathbf{k}_t \in \mathbb{R}^t$ is stretched into the degradation map $\mathcal{M} \in \mathbb{R}^{m\times n\times t}$. All values in the jth horizontal slice of $\mathcal{M}$ with the size $m \times n$ are the same as the jth element of $\mathbf{k}_t$. The degradation maps then can be concatenated with the detail patch, making CNN possible to handle the two inputs. It is noteworthy that this stretching operation is also in accordance with our assumption that rain streaks in a small patch are sharing the similar direction, and by doing so, every pixel of the input patch can receive the complete information of the blur kernel.

However, different from [41], in the case of plain motion blur kernel with only two parameters, we found that t is still too large compared with the number of channels of the detail patch. To balance the channel number of the detail patch and the degradation map, the degradation maps will be concatenated with the detail image after the first convolutional layer in the deraining network, as shown in Fig. 2.

D. Deraining Network

As previously mentioned in Section I, instead of elaborately designing the architecture, we resort to the typical ResNet structure. A cascade of "Conv + ReLU + BN" layers is

[1] As discussed in [40], the choice of the low-pass filters is not limited to the guided filter.

[2] Our parameter network learns the length and angle and has shown to be more simple and robust than learning all values of a motion kernel of size $p \times p$.

applied to perform deraining. Each layer consists of three types of operations, including convolution (denoted as "Conv"), rectified linear units [64] (denoted as "ReLU"), and batch normalization [65] (denoted as "BN"). We also adopt the Frobenius norm in the loss function, which is

$$L_{\mathrm{D}}(\Theta_{\mathrm{D}}) = \frac{1}{n}\sum_{i=1}^{n}\left\|\mathcal{F}_{\mathrm{D}}\left(\mathbf{O}_{\mathrm{T}}^{i}, \mathbf{K}^{i}; \Theta_{\mathrm{D}}\right) - \mathbf{R}^{i}\right\|_{F}^{2} \tag{4}$$

where $\mathbf{R}^i$ is the ith rain streaks patch. After subtracting the rain streaks $\mathbf{R}$ from the rainy image $\mathbf{O}$, we could get the rain-free image.

Discussions: As we mentioned earlier, distinguishing the rain streaks and the line pattern textures is important but challenging. In this work, we face this challenge by exploiting the generation mechanism of the rain streaks to guide the rain streaks removal. Within our framework, the generation mechanism of the rain streaks is taken into consideration, and the prior knowledge of the rain streaks, i.e., the angle and the length of the motion blur kernel, is automatically learned. Embedding of the motion blur kernel into the deraining network, which has a plain ResNet structure, greatly enhances the performance (see the comparisons in Section IV-D1).

IV. Experimental Results

To evaluate the performance of the proposed KGCNN framework, we test it on both synthetic and real-world rainy images. The networks are trained on synthesized rainy images. We compare the proposed KGCNN with four state-of-the-art methods, including two traditional methods: the layer prior (LP)[3] [20] and the unidirectional global sparse model (UGSM)[4] [16], as well as two deep learning-based methods: the DDN[5] [12] and the density-aware multistream deraining dense network (DID)[6] [11].

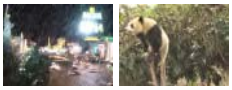
A. Experimental Settings

1) Rainy Images Simulation: With the observation model in (1), the synthetic rainy images are generated by the following steps.

1) Transform a clean image from RGB color space to YUV color space.[7]
2) Generate the raindrops mask $\mathbf{M}$ by adding salt and pepper noise with 1%–10% to a zero matrix with the same size as the Y channel of the clean image, and add a Gaussian blur with the standard variance from 0.2 to 0.5.
3) Generate the motion blur kernel $\mathbf{K}$ with the angle and length sampled from [45°, 135°] and [15, 30], respectively.
4) Directly add the generated rain streaks $\mathbf{R} = \mathbf{K} \otimes \mathbf{M}$ to the Y channel of the clean image, and the intensity values greater than 1 are set as 1.
5) Finally, transform the image back to RGB color space.

[3]http://yu-li.github.io/
[4]http://www.escience.cn/people/dengliangjian/index.html
[5]https://xueyangfu.github.io/projects/cvpr2017.html
[6]https://github.com/hezhangsprinter/DID-MDN
[7]https://en.wikipedia.org/wiki/YUV

TABLE I
Synthetic Data Sets

Dataset	# images	Details	Examples
No. 1	4	Rain streaks synthesised via motion blurring referred to [67, 68].	
No. 2	12	Generated by the authors of [20, 21] via the rendering technique [69].	
No. 3	200	Generated by the authors of [29].	

2) Implementation Details: For fair comparisons, we use the default parameters in the codes for traditional methods and the default trained models for the deep learning methods. Since existing rainy data sets do not consist of the information of the motion blur kernel, we train our networks only on our training data. Because of our assumption, the input image should be split into several patches for training and testing. The patch size is set as $64 \times 64 \times 3$ and the stride for selecting patches is set as 16. The radius of the guided filter for low-pass filtering is 15. By preserving 99% of the energy, the kernel is projected onto a space of 162 dimensions. We generated 1 million rainy/clean patch pairs with parameters $[\theta, l]$ for training and 500 rainy/clean patch pairs with parameters $[\theta, l]$ as the validation set from a data set containing 1000 clean images of [12]. The two (sub)networks are trained separately on the same data. We use Adam [66] optimizer with the learning rate 0.01. Our model is trained and tested on Python 3.5.2 with TensorFlow 1.0.1 framework on a desktop of GPU NVIDIA GeForce GTX 1060 with 6 GB. For other compared methods based on MATLAB, they are running on MATLAB 2017A. LP and UGSM are implemented on the CPU, whereas DID, DDN, and the proposed KGCNN are implemented on the GPU.

B. Experimental Results on Synthetic Data

In this section, we evaluate the performance of different state-of-the-art methods on synthetic rainy images. Here, we consider three synthetic data sets, which are summarized in Table I, as follows.

Data Set 1: Four synthetic rainy images (respectively, denoted as "night," "panda," "house," and "dog") simulated via motion blurring referred to [67] and [68]. The angle, with respect to the horizontal direction, of the rain streaks in "night" is fixed at $\theta = 45°$, while $\theta = 80°$ or $100°$ in "house." The rain streaks' angles are uniformly distributed in the range of [110°, 120°] in "panda" and [75°, 85°] $\bigcup$ [95°, 105°] in "dog." Data set 1 is designed mainly for testing the robustness of our method for various situations.

Data Set 2: Twelve synthetic rainy images[8] (denoted as "Rain12") simulated by using the photorealistic rendering technique [69] by Li *et al.* [20], [21].

[8]https://yu-li.github.io/

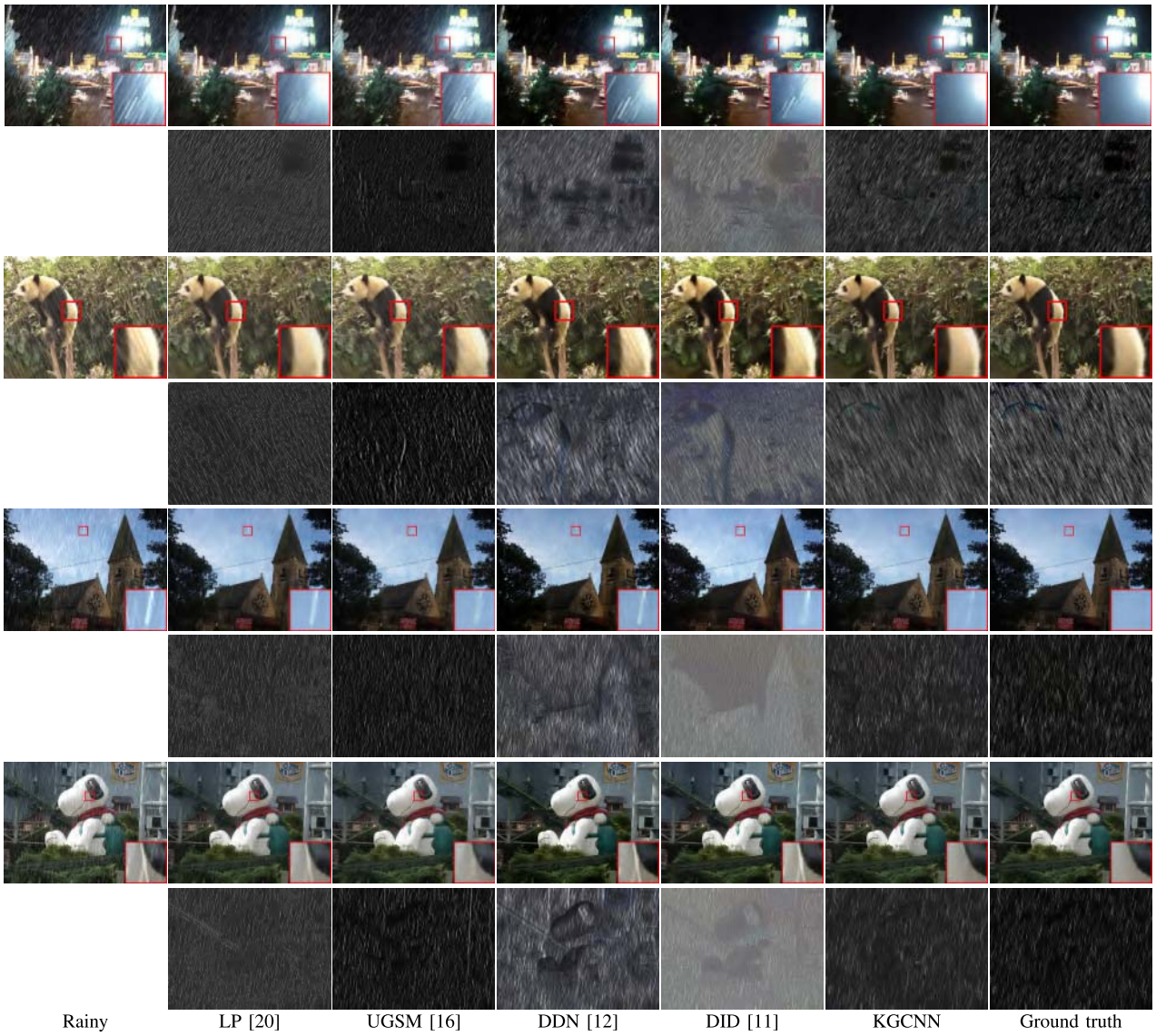


Fig. 4. From top to bottom, the rain streaks removal results by different methods on four rainy images in data set 1. For each image: the first row exhibits the rainy image, the rain streaks removal results by different methods, and the clean image; the second row displays the rain streaks images and the true rain streaks images.

Data Set 3: Two hundred rainy images[9] generated by Yang *et al.* [29]. Among them, 100 rainy images (denoted as "100L") are generated to simulate the light rain case, whereas the other 100 rainy images (denoted as "100H") are synthesized for the heavy rain case.

For quantitative comparisons, we adopt the peak signal-to-noise ratio (PSNR), structure similarity index (SSIM) [70], feature similarity index (FSIM) [71], universal image quality index (UIQI) [72], and gradient magnitude similarity deviation (GMSD; smaller is better) [73] as the quality metrics of the deraining results. We report the quality metrics of each image from data set 1. For saving space, we only give the average values for data sets 2 and 3.

Since the compared methods are implemented with different programming languages (e.g., UGSM was implemented with MATLAB, whereas DDN was implemented with Python), we first save the results by different methods as the.mat format without any compression and then reload the mat files and compute the corresponding quantitative metrics in the MATLAB for fairness. Meanwhile, to accurately and vividly illustrate the capabilities of different methods to remove the rain streaks and preserve the detail and the contrast, we exhibit both the deraining results and the rain streaks images. The rain streaks images are scaled for better visualization, and the scaling is identical for each rainy image.

The results on data set 1 are shown in Fig. 4 and Table II. From Fig. 4, it can be found that the proposed KGCNN is able to remove the rain streaks completely and preserves the details well. The compared methods tend to obtain either underderaining results, i.e., remaining obvious rain streaks or overderaining results, i.e., mistaken extraction of distinct structural information into the rain streaks images. For instance, DID removes almost all the rain streaks but usually brings a lot of content in the clean image into the rain streaks images. UGSM favors separating vertical line patterns and this limits its applicability for different scenarios.

[9]http://www.icst.pku.edu.cn/struct/Projects/joint_rain_removal.html

TABLE II

QUANTITATIVE COMPARISONS OF RAIN STREAKS REMOVAL RESULTS BY DIFFERENT METHODS ON DATA SET 1. THE BEST QUANTITATIVE VALUES ARE HIGHLIGHTED IN BOLDFACE

Image	Method	PSNR	SSIM	FSIM	UIQI	GMSD	Time (s)
night $\theta = 45°$	rainy	18.506	0.539	0.787	0.753	0.261	-
	LP [20]	22.984	0.777	0.872	0.806	0.175	511.492
	UGSM [16]	20.626	0.641	0.824	0.784	0.225	1.207
	DDN [12]	23.833	0.679	0.882	0.767	0.173	0.281
	DID [11]	23.996	0.828	0.924	0.725	0.097	**0.219**
	KGCNN	**31.415**	**0.957**	**0.970**	**0.929**	**0.046**	3.780
panda $\theta \in [110°, 120°]$	rainy	17.569	0.750	0.871	0.836	0.140	-
	LP [20]	21.188	0.857	0.905	0.907	0.104	103.666
	UGSM [16]	19.621	0.798	0.876	0.885	0.118	0.401
	DDN [12]	23.210	0.867	0.904	0.971	0.099	**0.078**
	DID [11]	23.842	0.879	0.909	0.962	0.095	0.219
	KGCNN	**27.060**	**0.927**	**0.938**	**0.980**	**0.086**	1.172
house $\theta \in \{80°, 100°\}$	rainy	23.265	0.766	0.833	0.831	0.208	-
	LP [20]	29.129	0.898	0.937	0.894	0.099	278.365
	UGSM [16]	29.368	0.911	0.943	0.898	0.089	1.077
	DDN [12]	28.687	0.869	0.933	0.861	0.097	**0.687**
	DID [11]	26.401	0.876	0.939	**0.945**	0.088	1.280
	KGCNN	**32.101**	**0.940**	**0.965**	0.926	**0.064**	2.999
dog $\theta \in \{[75°, 85°], [95°, 105°]\}$	rainy	22.643	0.729	0.886	0.919	0.187	-
	LP [20]	28.330	0.891	0.937	0.963	0.100	320.849
	UGSM [16]	28.919	0.908	0.944	0.961	0.089	1.337
	DDN [12]	26.097	0.847	0.931	0.919	0.111	**0.141**
	DID [11]	26.794	0.846	0.935	0.968	0.096	0.234
	KGCNN	**32.773**	**0.948**	**0.970**	**0.986**	**0.057**	2.765
average	rainy	20.496	0.696	0.844	0.835	0.199	-
	LP [20]	25.408	0.856	0.913	0.893	0.120	303.593
	UGSM [16]	24.634	0.814	0.897	0.882	0.130	1.005
	DDN [12]	25.457	0.815	0.913	0.879	0.120	**0.297**
	DID [11]	25.258	0.857	0.927	0.900	0.094	0.488
	KGCNN	**30.837**	**0.943**	**0.960**	**0.955**	**0.063**	2.679

TABLE III

QUANTITATIVE COMPARISONS (AVERAGE VALUES) OF RAIN STREAKS REMOVAL RESULTS BY DIFFERENT METHODS ON DATA SET 2. THE BEST QUANTITATIVE VALUES ARE HIGHLIGHTED IN BOLDFACE

Method	PSNR	SSIM	FSIM	UIQI	GMSD	Time (s)
rainy	28.822	0.910	0.910	0.968	0.134	-
LP [20]	30.833	0.947	0.935	0.968	0.070	338.005
UGSM [16]	32.185	0.958	0.947	0.983	0.065	0.944
DDN [12]	29.421	0.938	0.942	0.939	0.073	**0.164**
DID [11]	27.040	0.902	0.909	0.972	0.088	0.221
KGCNN	**34.907**	**0.975**	**0.969**	**0.990**	**0.048**	2.856

It is notable that the images in data set 1 consist of abundant line textures, and we can find from Fig. 4 that DDN suffers from the difficulty of distinguishing the line pattern textures and the rain streaks. In contrast, the rain streaks images in Fig. 4 extracted by the proposed KGCNN is the closest to the simulated rain streaks, and this indicates that our method could distinguish the line pattern textures and the rain streaks well.

Table III exhibits the quantitative comparisons of the results by different methods on the 12 images in the data set 2, i.e., "Rain12." The PSNR of KGCNN is the highest and the margins are more than 2.7 dB. The proposed KGCNN also achieves the best SSIM, FSIM, UIQI, and GMSD. Fig. 5 shows the results of different methods on four images in data set 2. From the zoomed-in areas, it can be observed that the proposed KGCNN removes the rain streaks, whereas other compared methods fail to do so. We can see obvious structural information in the extracted rain streaks images by the compared methods, except UGSM, in Fig. 5. The rain streaks image by the proposed KGCNN is the closest to the true rain streaks and does not contain details of the clean image. This demonstrates that the proposed KGCNN preserves the details very well.

Table IV shows the quantitative comparisons of the results by different methods on data set 3. Except for the UIQI and GMSD, the proposed KGCNN achieves the best quantitative metrics. Fig. 6 shows the results of different methods on four images in data set 3. It can be observed that the proposed KGCNN removes the rain streaks, whereas other compared methods fail to do so. The rain streaks image by the proposed KGCNN is the closest to the true rain streaks and does not contain structures of the details. This demonstrates that the proposed KGCNN preserves the details very well too.

The running time of different methods is also reported in Tables II–IV. All the methods are very fast except LP. Although our KGCNN is slower than DID and DDN because of its patchwise operation, it achieves the best quantitative metrics in Tables II–IV.

C. Experimental Results on Real-World Data

In this section, we exhibit the results on real-world rainy data. Seven images are selected in our experiment. The first two are selected from the rain streaks removal literature [12], [21], respectively, whereas the other five are downloaded from the Internet. We display the deraining images and the corresponding rain streaks images in Fig. 7.

From the top six rows of Fig. 7 (corresponding to the first three rainy images), we can observe that DDN, DID, and the proposed KGCNN obtain similar visual results and remove rain streaks completely, whereas LP fails to remove all rain streaks and UGSM remains a lot of artifacts. We can find that DID and DDN fail to separate the rain streaks and details well and extract some details (e.g., the grasses in the zoomed-in area of the second rainy image and the truck with the vertical and white appearance in the zoomed-in area of the third rainy image) to rain streaks, which demonstrates that DID and DDN could not distinguish details and the rain streaks well.

The rain streaks, in the fourth and fifth rainy images of Fig. 7, are of different directions. From the zoomed-in area in the seventh row, we can observe that our method removes the rain streaks well, whereas compared methods remain obvious rain streaks. Although UGSM and our KGCNN remove almost all the rain streaks, performing better than the other three methods, on the fifth rainy image, we can observe from the rain streak images that UGSM incorrectly removes the line patterns on the red dress of the girl. The similar phenomena of this improper removal also appear in the rain streak images by LP and DID in the ninth or tenth row of Fig. 7.

The results on the sixth and seventh rainy images (11–14 rows in Fig. 7) also illustrate that our KGCNN can remove all the rain streaks and preserve the background well. In Fig. 7, another direct-viewing evidence of our method preserving background well is that we can hardly identify the scene from the rain streak images by KGCNN, while we can readily recognize human beings or roads in the rain streak images by compared methods. It is noteworthy that the preservation of

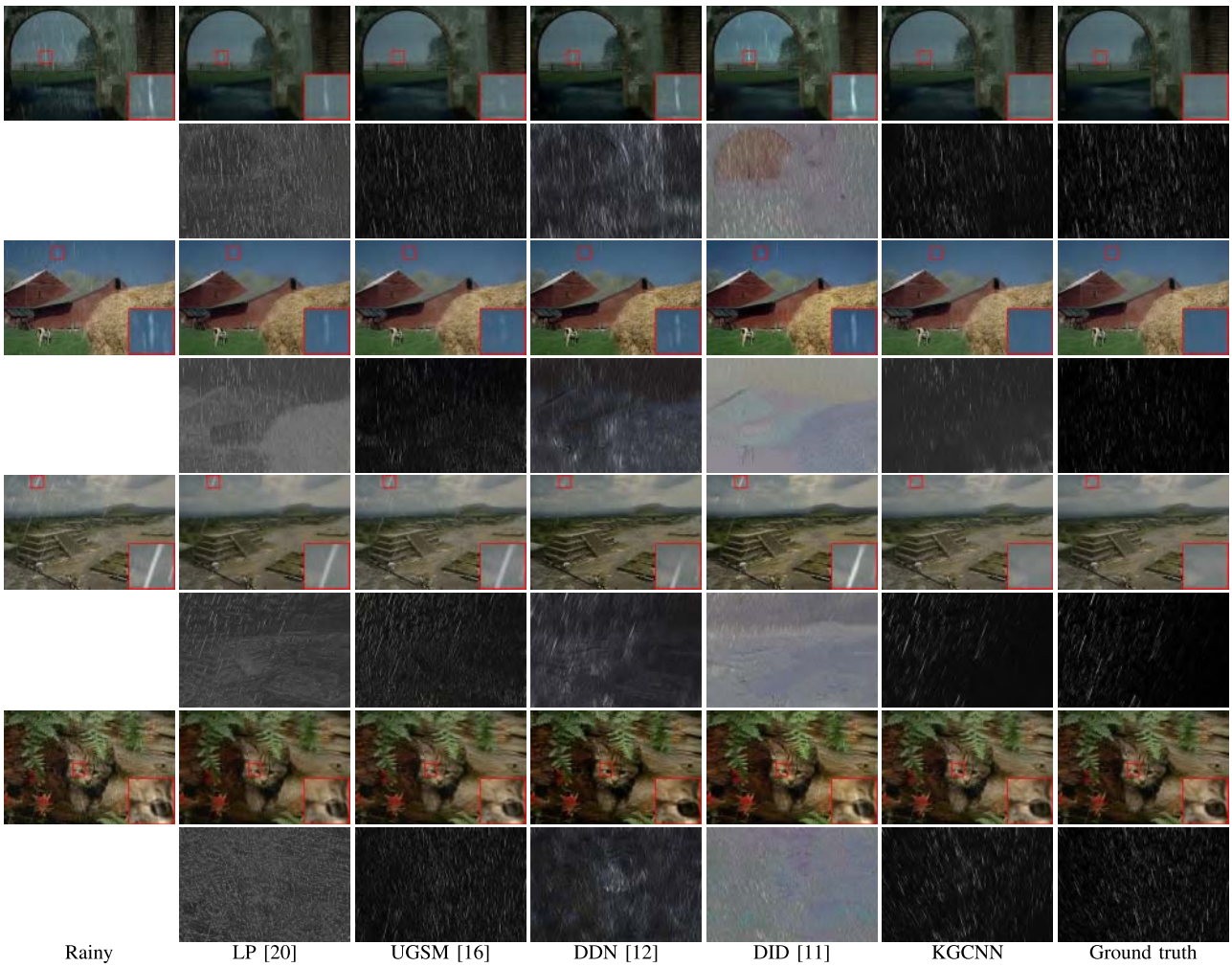


Fig. 5. From top to bottom, rain streaks removal results by different methods on four rainy images in data set 2. For each image: the first row exhibits the rainy image, the rain streaks removal results by different methods, and the clean image; the second row displays the rain streaks images and the true rain streaks images.

TABLE IV

QUANTITATIVE COMPARISONS (AVERAGE VALUES) OF RAIN STREAKS REMOVAL RESULTS BY DIFFERENT METHODS ON DATA SET 3. THE BEST QUANTITATIVE VALUES ARE HIGHLIGHTED IN BOLDFACE

Images	Method	PSNR	SSIM	FSIM	UIQI	GMSD	Time (s)
100L	rainy	25.336	0.903	0.885	0.967	0.178	-
	LP [20]	27.533	0.929	0.907	0.971	0.124	404.809
	UGSM [16]	28.918	0.946	0.926	**0.982**	0.110	0.947
	DDN [12]	25.894	0.918	0.910	0.945	0.129	**0.117**
	DID [11]	23.869	0.880	0.884	0.920	0.138	0.263
	KGCNN	**29.953**	**0.959**	**0.944**	0.981	**0.105**	2.806
100H	rainy	11.678	0.484	0.619	0.732	0.292	-
	LP [20]	13.234	0.551	0.662	0.771	0.277	340.091
	UGSM [16]	14.804	0.653	0.732	0.803	0.237	1.894
	DDN [12]	14.735	0.590	0.676	0.828	0.266	**0.115**
	DID [11]	15.373	0.679	0.743	0.836	**0.233**	0.242
	KGCNN	**17.774**	**0.720**	**0.759**	**0.883**	0.240	2.373

the background is of great importance when we position the rain streaks removal approaches at the preprocessing stage. Incorrect removal may cause serious consequences in the applications, the automatic driving technique for example.

D. Discussion

1) Influence of the Motion Blur Kernel: In this article, we propose a KGCNN method for the single image rain streaks removal application. The motion blur kernel plays a vitally important role both in our generation mechanism and the deraining framework. However, there are still two uncertainties: 1) how is the performance if we directly use the deraining network, which is trained regularly in our framework, without the kernel information? 2) how is the performance if we independently train the deraining network with the rainy and clean image pairs?

To show the significance of the motion blur kernel in our method, we do the ablation analysis on data set 2. We use KGCNN^{a} to denote the proposed method with the kernel being zeros, i.e., without the kernel information, and KGCNN^{b} to represent the deraining network trained independently with our training data. It is notable that the structure of KGCNN^{b} is very similar to that of DDN. Fig. 8 shows the visual results of three images in data set 2. It is not difficult to see that the kernel plays an important role in KGCNN. The performance of KGCNN^{a} and KGCNN^{b} is not desirable. The quantitative results in Table V also demonstrate the same conclusion. In summary, the motion blur kernel is very important in the framework of the proposed KGCNN method.

Next, we discuss the influence of the directionality at the deraining stage. The information of the rain streaks' direction would help the deraining network to specify the particular

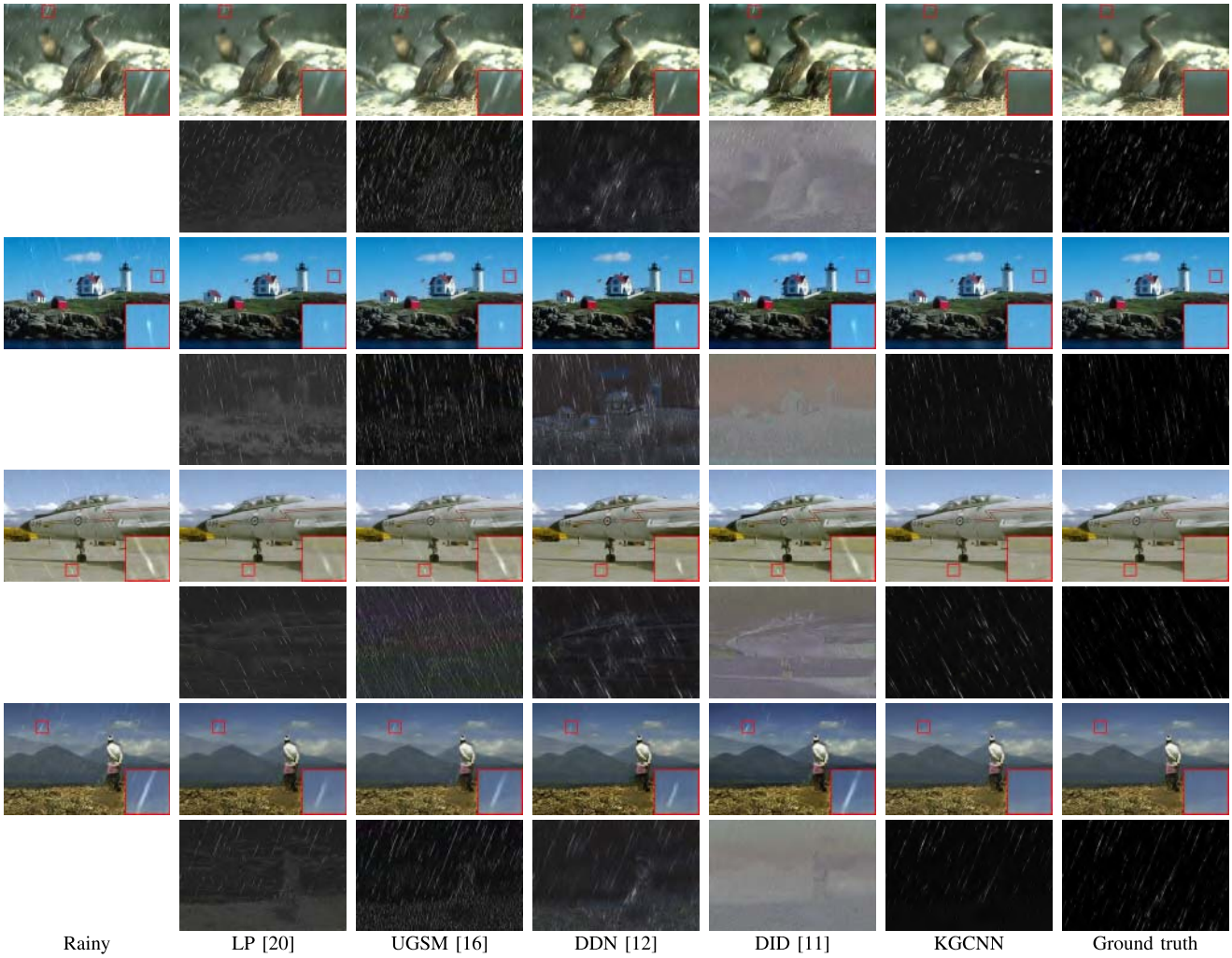


Fig. 6. From top to bottom, rain streaks removal results by different methods on four rainy images in data set 3. For each image, the first row exhibits the rainy image, the rain streaks removal results by different methods, and the clean image; the second row displays the rain streaks images and the ground truth rain streaks.

TABLE V

QUANTITATIVE COMPARISONS (AVERAGE VALUE) OF RAIN STREAKS REMOVAL RESULTS BY KGCNN^a, KGCNN^b, AND THE PROPOSED KGCNN ON DATA SET 2. THE BEST QUANTITATIVE VALUES ARE HIGHLIGHTED IN BOLDFACE

Method	PSNR	SSIM	FSIM	UIQI	GMSD	Time (s)
rainy	28.822	0.910	0.910	0.968	0.134	-
KGCNN^a	29.668	0.924	0.919	0.972	0.126	2.501
KGCNN^b	33.154	0.960	0.950	0.986	0.065	**0.269**
KGCNN	**34.907**	**0.975**	**0.969**	**0.990**	**0.048**	2.856

TABLE VI

QUANTITATIVE COMPARISONS OF RAIN STREAKS REMOVAL RESULTS BY KGCNN WITH INCORRECT ANGLES OF MOTION BLUR KERNELS ON DATA SET 2. THE BEST QUANTITATIVE VALUES ARE HIGHLIGHTED IN BOLDFACE

Δ_θ	PSNR	SSIM	FSIM	UIQI	GMSD
-10°	31.107	0.943	0.934	0.970	0.107
-5°	33.600	0.966	0.958	**0.996**	0.068
0	**34.907**	**0.975**	**0.969**	0.990	**0.048**
5°	32.640	0.957	0.950	0.989	0.081
10°	30.423	0.933	0.927	0.968	0.114

target. To clarify the role of the direction property, we conduct the experiments, in which the input motion blur kernels are with incorrect angles. The results are shown in Table VI. Δ_θ is the difference value between the angle estimated by our parameter network and the angle of the motion blur kernel input into the deraining network. From the quantitative metrics in Table VI, we can conclude that the directionality is crucial in our framework and the estimation of the parameter network is accurate. This also reveals the reason why our method preserves the vertical line pattern of the grass well in the zoomed-in red boxes in Fig. 1.

2) Parameters: In our framework, some preset parameters related to the network structure, e.g., the depth and width of the ResNet backbone and the stride in patchwise deraining, would affect the performance and running time. To figure out their impacts, we conduct experiments with different parameter settings. Since that the parameter network could accurately estimate the motion blur kernel with a simple structure, we omit the discussion on the parameters of its structure.

As for the deraining network, we test our method on data set 2 by setting the network depth as 18, 26, and 34 and the network width as 24, 36, and 48. The depth indicates the number of "Conv + ReLU + BN" layers shown in Fig. 2. The width is the number of the convolutional filers in each layer. Table VII shows the average values of the quantitative results. It can be seen that our method achieves good performances

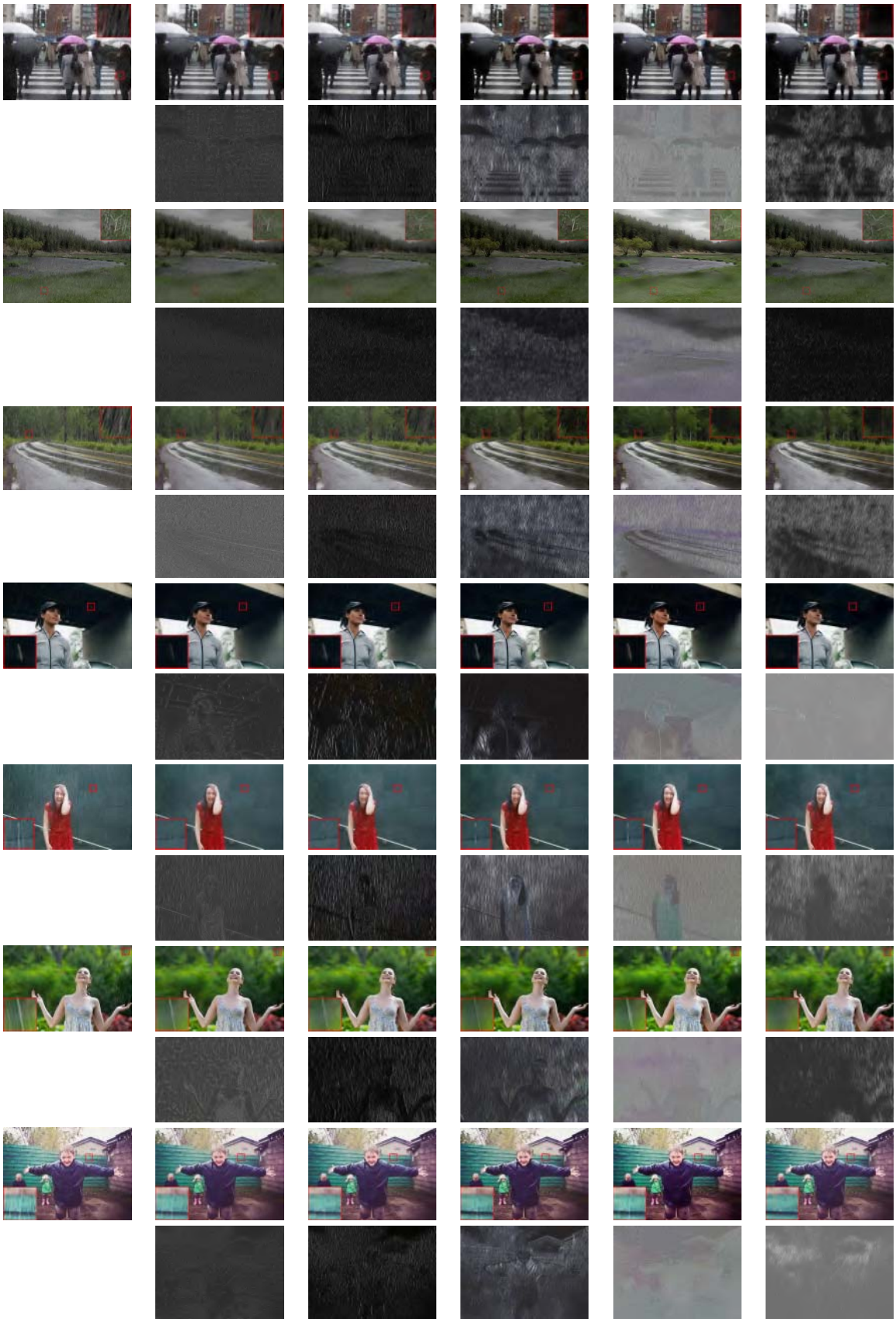


Fig. 7. From top to bottom, rain streaks removal results by different methods on real rainy images. For each image, the first row exhibits the rainy image and the rain streaks removal results by different methods; the second row displays the rain streaks images.

Fig. 8. Rain streaks removal results by KGCNN[a], KGCNN[b], and the proposed KGCNN on three rainy images in data set 2.

TABLE VII

QUANTITATIVE COMPARISONS (AVERAGE VALUE) OF RAIN STREAKS REMOVAL RESULTS WITH DIFFERENT DEPTHS (# LAYERS) AND WIDTHS (# FILTERS) ON DATA SET 2. THE BEST QUANTITATIVE VALUES ARE HIGHLIGHTED IN BOLDFACE

# layers	# filter	PSNR	SSIM	FSIM	UIQI	GMSD	Time (s)
18	24	34.252	0.973	0.967	0.979	0.050	**1.938**
	36	34.972	0.977	0.970	**0.992**	0.045	2.393
	48	35.251	0.977	0.971	0.988	0.044	2.655
26	24	34.936	0.976	0.970	0.986	0.045	2.252
	36	34.907	0.975	0.969	0.990	0.048	2.856
	48	**35.403**	**0.978**	**0.972**	0.990	0.044	3.333
34	24	35.335	0.977	0.971	0.991	**0.043**	2.592
	36	35.123	0.977	0.971	0.985	0.044	3.477
	48	35.203	0.975	0.969	0.981	0.049	3.999

TABLE VIII

QUANTITATIVE COMPARISONS (AVERAGE VALUE) OF RAIN STREAKS REMOVAL RESULTS OF THE PROPOSED METHOD WITH DIFFERENT STRIDES ON DATA SET 2. THE BEST QUANTITATIVE VALUES ARE HIGHLIGHTED IN BOLDFACE

Stride	PSNR	SSIM	FSIM	UIQI	GMSD	Time (s)
16	**34.907**	**0.975**	**0.969**	**0.990**	**0.048**	2.856
32	34.782	0.975	0.969	0.984	0.049	0.896
48	34.634	0.974	0.968	0.985	0.051	**0.473**

with respect to different settings of the network depth and width, showing robustness in the selected ranges. Generally, increasing the number of layers and the number of filters in each layer generally help the network to achieve a better representation ability but increase the computational time. However, we could see that when the depth is 34 and the filter number is 48, the PSNR decreases. This may be caused by overfitting, i.e., the network could not deal with the data that are different from the training data. To balance the performance and the computation time, we set the layer number as 26 and the filter number as 36 in the "deraining network".

Finally, we test our method on data set 2 by setting the stride as 16, 32, and 48. For an image of size $M \times N$, the number of patches whose size is $m \times n$ for the stride s is $\lceil((M-m+1)/s)\rceil \times \lceil((N-n+1)/s)\rceil$, where $\lceil\cdot\rceil$ is the ceiling function. Table VIII reports the quantitative metrics and the running time with respect to different strides. Since the number of patches increases inversely with the stride, we can observe that the proposed method could run faster with a small loss of performance metrics. Meanwhile, the running time is also acceptable when $s = 16$. Therefore, for a better performance of the proposed KGCNN, we set the default stride as 16 in our experiment.

V. CONCLUSION

We present a deep learning architecture called KGCNN for removing rain streaks in single rainy images. With the guidance of the motion blur kernel, our approach learns the mapping function between the rainy image on the detail layer and the rain streaks on the detail layer. We show that the simple CNN guided by the generation mechanism can tackle natural images under bad weather conditions successfully. The extensive experimental results also validate that the proposed KGCNN noticeably outperforms state-of-the-art methods qualitatively and quantitatively. In the future work, we can develop our method in a recursive manner as in [32]–[34] to flexibly handle very complex rainy scenarios.

ACKNOWLEDGMENT

The authors would like to express their sincere thanks to the editor and referees for giving them so many valuable comments and suggestions for revising this article. They would like to thank the authors of DID [11], LP [20], and DDN [12] for providing the code and for the support form Financial Intelligence and Financial Engineering Research Key Laboratory of Sichuan Province.

REFERENCES

[1] K. Garg and S. K. Nayar, "Vision and rain," *Int. J. Comput. Vis.*, vol. 75, no. 1, pp. 3–27, Jul. 2007.
[2] X. Zhang, C. Zhu, S. Wang, Y. Liu, and M. Ye, "A Bayesian approach to camouflaged moving object detection," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 27, no. 9, pp. 2001–2013, Sep. 2017.
[3] S.-C. Huang and B.-H. Chen, "Highly accurate moving object detection in variable bit rate video-based traffic monitoring systems," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 24, no. 12, pp. 1920–1931, Dec. 2013.
[4] Z. Ma, X. Chang, Z. Xu, N. Sebe, and A. G. Hauptmann, "Joint attributes and event analysis for multimedia event detection," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 7, pp. 2921–2930, Jul. 2018.
[5] S. Song, C. Lan, J. Xing, W. Zeng, and J. Liu, "Spatio-temporal attention-based LSTM networks for 3D action recognition and detection," *IEEE Trans. Image Process.*, vol. 27, no. 7, pp. 3459–3471, Jul. 2018.

[6] H. Liu, N. Shu, Q. Tang, and W. Zhang, "Computational model based on neural network of visual cortex for human action recognition," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 5, pp. 1427–1440, May 2018.

[7] L. Itti, C. Koch, and E. Niebur, "A model of saliency-based visual attention for rapid scene analysis," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 20, no. 11, pp. 1254–1259, 1998.

[8] Y. Yuan, L. Mou, and X. Lu, "Scene recognition by manifold regularized deep learning architecture," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 26, no. 10, pp. 2222–2233, Oct. 2015.

[9] C. H. Bahnsen and T. B. Moeslund, "Rain removal in traffic surveillance: Does it matter?" *IEEE Trans. Intell. Transp. Syst.*, vol. 20, no. 8, pp. 2802–2819, Aug. 2019.

[10] S. Li *et al.*, "Single image deraining: A comprehensive benchmark analysis," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 3833–3842.

[11] H. Zhang and V. M. Patel, "Density-aware single image de-raining using a multi-stream dense network," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 695–704.

[12] X. Fu, J. Huang, D. Zeng, Y. Huang, X. Ding, and J. Paisley, "Removing rain from single images via a deep detail network," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 1715–1723.

[13] L.-W. Kang, C.-W. Lin, and Y.-H. Fu, "Automatic single-image-based rain streaks removal via image decomposition," *IEEE Trans. Image Process.*, vol. 21, no. 4, pp. 1742–1755, Apr. 2012.

[14] S.-H. Sun, S.-P. Fan, and Y.-C.-F. Wang, "Exploiting image structural similarity for single image rain removal," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Oct. 2014, pp. 4482–4486.

[15] Y. Chang, L. Yan, and S. Zhong, "Transformed low-rank model for line pattern noise removal," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 1735–1743.

[16] L.-J. Deng, T.-Z. Huang, X.-L. Zhao, and T.-X. Jiang, "A directional global sparse model for single image rain removal," *Appl. Math. Model.*, vol. 59, pp. 662–679, Jul. 2018.

[17] S. Du, Y. Liu, M. Ye, Z. Xu, J. Li, and J. Liu, "Single image deraining via decorrelating the rain streaks and background scene in gradient domain," *Pattern Recognit.*, vol. 79, pp. 303–317, Jul. 2018.

[18] T.-X. Jiang, T.-Z. Huang, X.-L. Zhao, L.-J. Deng, and Y. Wang, "A novel tensor-based video rain streaks removal approach via utilizing discriminatively intrinsic priors," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 4057–4066.

[19] Y.-L. Chen and C.-T. Hsu, "A generalized low-rank appearance model for spatio-temporally correlated rain streaks," in *Proc. IEEE Int. Conf. Comput. Vis.*, Dec. 2013, pp. 1968–1975.

[20] Y. Li, R. T. Tan, X. Guo, J. Lu, and M. S. Brown, "Rain streak removal using layer priors," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 2736–2744.

[21] Y. Li, R. T. Tan, X. Guo, J. Lu, and M. S. Brown, "Single image rain streak decomposition using layer priors," *IEEE Trans. Image Process.*, vol. 26, no. 8, pp. 3874–3885, Aug. 2017.

[22] Y. Luo, Y. Xu, and H. Ji, "Removing rain from a single image via discriminative sparse coding," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Dec. 2015, pp. 3397–3405.

[23] W. Wei, L. Yi, Q. Xie, Q. Zhao, D. Meng, and Z. Xu, "Should we encode rain streaks in video as deterministic or stochastic?" in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 2535–2544.

[24] M. Li *et al.*, "Video rain streak removal by multiscale convolutional sparse coding," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 6644–6653.

[25] S. Gu, D. Meng, W. Zuo, and L. Zhang, "Joint convolutional analysis and synthesis sparse representation for single image layer separation," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 1717–1725.

[26] H. Zhang and V. M. Patel, "Convolutional sparse and low-rank coding-based rain streak removal," in *Proc. IEEE Winter Conf. Appl. Comput. Vis. (WACV)*, Mar. 2017, pp. 1259–1267.

[27] R. Qian, R. T. Tan, W. Yang, J. Su, and J. Liu, "Attentive generative adversarial network for raindrop removal from a single image," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 2482–2491.

[28] J. Chen, C.-H. Tan, J. Hou, L.-P. Chau, and H. Li, "Robust video content alignment and compensation for rain removal in a CNN framework," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 6286–6295.

[29] W. Yang, R. T. Tan, J. Feng, J. Liu, Z. Guo, and S. Yan, "Deep joint rain detection and removal from a single image," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 1357–1366.

[30] J. Liu, W. Yang, S. Yang, and Z. Guo, "Erase or fill? Deep joint recurrent rain removal and reconstruction in videos," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 3233–3242.

[31] J. Liu, W. Yang, S. Yang, and Z. Guo, "D3R-Net: Dynamic routing residue recurrent network for video rain removal," *IEEE Trans. Image Process.*, vol. 28, no. 2, pp. 699–712, Feb. 2019.

[32] X. Li, J. Wu, Z. Lin, H. Liu, and H. Zha, "Recurrent squeeze-and-excitation context aggregation net for single image deraining," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, Sep. 2018, pp. 254–269.

[33] W. Yang, R. T. Tan, J. Feng, Z. Guo, S. Yan, and J. Liu, "Joint rain detection and removal from a single image with contextualized deep networks," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 42, no. 6, pp. 1377–1393, Jun. 2020.

[34] D. Ren, W. Zuo, Q. Hu, P. Zhu, and D. Meng, "Progressive image deraining networks: A better and simpler baseline," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 3932–3941.

[35] T. Wang, X. Yang, K. Xu, S. Chen, Q. Zhang, and R. W. H. Lau, "Spatial attentive single-image deraining with a high quality real rain dataset," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 12262–12271.

[36] W. Wei, D. Meng, Q. Zhao, Z. Xu, and Y. Wu, "Semi-supervised transfer learning for image rain removal," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 3872–3881.

[37] R. Li, L.-F. Cheong, and R. T. Tan, "Heavy rain image restoration: Integrating physics model and conditional adversarial learning," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 1633–1642.

[38] W. Yang, J. Liu, and J. Feng, "Frame-consistent recurrent video deraining with dual-level flow," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 1661–1670.

[39] X. Hu, C.-W. Fu, L. Zhu, and P.-A. Heng, "Depth-attentional features for single-image rain removal," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 8014–8023.

[40] X. Fu, J. Huang, X. Ding, Y. Liao, and J. Paisley, "Clearing the skies: A deep network architecture for single-image rain removal," *IEEE Trans. Image Process.*, vol. 26, no. 6, pp. 2944–2956, Jun. 2017.

[41] K. Zhang, W. Zuo, and L. Zhang, "Learning a single convolutional super-resolution network for multiple degradations," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 3262–3271.

[42] B.-H. Chen, S.-C. Huang, C.-Y. Li, and S.-Y. Kuo, "Haze removal using radial basis function networks for visibility restoration applications," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 8, pp. 3828–3838, Aug. 2018.

[43] R. Liu, X. Fan, M. Hou, Z. Jiang, Z. Luo, and L. Zhang, "Learning aggregated transmission propagation networks for haze removal and beyond," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 30, no. 10, pp. 2973–2986, Oct. 2019.

[44] J.-H. Kim, C. Lee, J.-Y. Sim, and C.-S. Kim, "Single-image deraining using an adaptive nonlocal means filter," in *Proc. IEEE Int. Conf. Image Process.*, Sep. 2013, pp. 914–917.

[45] L. Zhu, C.-W. Fu, D. Lischinski, and P.-A. Heng, "Joint bi-layer optimization for single-image rain streak removal," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 2545–2553.

[46] D. Ren, W. Zuo, D. Zhang, L. Zhang, and M.-H. Yang, "Simultaneous fidelity and regularization learning for image restoration," *IEEE Trans. Pattern Anal. Mach. Intell.*, early access, Jul. 2, 2019, doi: 10.1109/TPAMI.2019.2926357.

[47] Y. Wang, S. Liu, C. Chen, and B. Zeng, "A hierarchical approach for rain or snow removing in a single color image," *IEEE Trans. Image Process.*, vol. 26, no. 8, pp. 3936–3950, Aug. 2017.

[48] K. Garg and S. K. Nayar, "Detection and removal of rain from videos," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. CVPR*, Jun. 2004, p. 1.

[49] A. K. Tripathi and S. Mukhopadhyay, "Video post processing: Low-latency spatiotemporal approach for detection and removal of rain," *IET Image Process.*, vol. 6, no. 2, pp. 181–196, Mar. 2012.

[50] J. Chen and L.-P. Chau, "A rain pixel recovery algorithm for videos with highly dynamic scenes," *IEEE Trans. Image Process.*, vol. 23, no. 3, pp. 1097–1104, Mar. 2014.

[51] J.-H. Kim, J.-Y. Sim, and C.-S. Kim, "Video deraining and desnowing using temporal correlation and low-rank matrix completion," *IEEE Trans. Image Process.*, vol. 24, no. 9, pp. 2658–2670, Sep. 2015.

[52] V. Santhaseelan and V. K. Asari, "Utilizing local phase information to remove rain from video," *Int. J. Comput. Vis.*, vol. 112, no. 1, pp. 71–89, Mar. 2015.

[53] A. K. Tripathi and S. Mukhopadhyay, "Removal of rain from videos: A review," *Signal, Image Video Process.*, vol. 8, no. 8, pp. 1421–1430, Nov. 2014.

[54] S. You, R. T. Tan, R. Kawakami, Y. Mukaigawa, and K. Ikeuchi, "Adherent raindrop modeling, detection and removal in video," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 38, no. 9, pp. 1721–1733, Sep. 2016.

[55] W. Ren, J. Tian, Z. Han, A. Chan, and Y. Tang, "Video desnowing and deraining based on matrix decomposition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 4210–4219.

[56] D. Eigen, D. Krishnan, and R. Fergus, "Restoring an image taken through a window covered with dirt or rain," in *Proc. IEEE Int. Conf. Comput. Vis.*, Dec. 2013, pp. 633–640.

[57] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778.

[58] J. Pan *et al.*, "Learning dual convolutional neural networks for low-level vision," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 3070–3079.

[59] X. Fu, B. Liang, Y. Huang, X. Ding, and J. Paisley, "Lightweight pyramid networks for image deraining," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 31, no. 6, pp. 1794–1807, Jun. 2020.

[60] W. Yang, J. Liu, S. Yang, and Z. Guo, "Scale-free single image deraining via visibility-enhanced recurrent wavelet learning," *IEEE Trans. Image Process.*, vol. 28, no. 6, pp. 2948–2961, Jun. 2019.

[61] W. Yang, S. Wang, D. Xu, X. Wang, and J. Liu, "Towards scale-free rain streak removal via self-supervised fractal band learning," in *Proc. AAAI Conf. Artif. Intell. (AAAI)*, 2020, pp. 12629–12636.

[62] D. Ren, W. Shang, P. Zhu, Q. Hu, D. Meng, and W. Zuo, "Single image deraining using bilateral recurrent network," *IEEE Trans. Image Process.*, vol. 29, pp. 6852–6863, 2020.

[63] K. He, J. Sun, and X. Tang, "Guided image filtering," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 6, pp. 1397–1409, Jun. 2013.

[64] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, 2012, pp. 1097–1105.

[65] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *Proc. Int. Conf. Mach. Learn. (ICML)*, 2015, pp. 448–456.

[66] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014, *arXiv:1412.6980.* [Online]. Available: http://arxiv.org/abs/1412.6980

[67] T.-X. Jiang, T.-Z. Huang, X.-L. Zhao, L.-J. Deng, and Y. Wang, "FastDeRain: A novel video rain streak removal method using directional gradient priors," *IEEE Trans. Image Process.*, vol. 28, no. 4, pp. 2089–2102, Apr. 2019.

[68] Y.-T. Wang *et al.*, "A total variation and group sparsity based tensor optimization model for video rain streak removal," *Signal Process., Image Commun.*, vol. 73, pp. 96–108, Apr. 2019.

[69] K. Garg and S. K. Nayar, "Photorealistic rendering of rain streaks," *ACM Trans. Graph.*, vol. 25, no. 3, pp. 996–1002, Jul. 2006.

[70] 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.

[71] L. Zhang, L. Zhang, X. Mou, and D. Zhang, "FSIM: A feature similarity index for image quality assessment," *IEEE Trans. Image Process.*, vol. 20, no. 8, pp. 2378–2386, Aug. 2011.

[72] Z. Wang and A. C. Bovik, "A universal image quality index," *IEEE Signal Process. Lett.*, vol. 9, no. 3, pp. 81–84, Mar. 2002.

[73] W. Xue, L. Zhang, X. Mou, and A. C. Bovik, "Gradient magnitude similarity deviation: A highly efficient perceptual image quality index," *IEEE Trans. Image Process.*, vol. 23, no. 2, pp. 684–695, Feb. 2014.

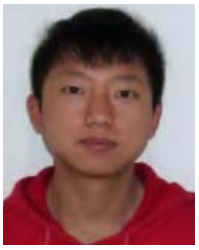
Ye-Tao Wang received the B.S. and M.S. degrees from the University of Electronic Science and Technology of China (UESTC), Chengdu, China, in 2016 and 2019, with a focus on rain streaks removal, where he is currently pursuing the Ph.D. degree with the School of Mathematical Sciences.

Xi-Le Zhao (Member, IEEE) received the M.S. and Ph.D. degrees from the University of Electronic Science and Technology of China (UESTC), Chengdu, China, in 2009 and 2012, respectively.

He is currently a Professor with the School of Mathematical Sciences, UESTC. His research interests include model- and data-driven methods for image processing problems. His homepage is https://zhaoxile.github.io/.

Tai-Xiang Jiang (Member, IEEE) received the B.S. and Ph.D. degrees in mathematics and applied mathematics from the University of Electronic Science and Technology of China (UESTC), Chengdu, China, in 2013.

He is currently an Associate Professor with the School of Economic Information Engineering, Southwestern University of Finance and Economics, Chengdu. His research interests include sparse and low-rank modeling, tensor decomposition, and multidimensional image processing. https://sites.google.com/view/taixiangjiang/



Liang-Jian Deng (Member, IEEE) received the B.S. and Ph.D. degrees from the School of Mathematical Sciences, University of Electronic Science and Technology of China (UESTC), Chengdu, China, in 2010 and 2016, respectively.

From September 2013 to September 2014, he stayed at Case Western Reserve University (CWRU), Cleveland, OH, USA, as a joint-training Ph.D. Student. In 2017, he held a post-doctoral position at Hong Kong Baptist University (HKBU), Hong Kong. He also stayed at the Isaac Newton Institute for Mathematical Sciences, University of Cambridge, Cambridge, U.K., and HKBU for short visits. He is currently an Associate Professor with the School of Mathematical Sciences, UESTC. His research interest is mainly focusing on using partial differential equation (PDE), optimization modeling, and deep learning to address several tasks in image processing and computer vision, e.g., resolution enhancement and restoration.

Yi Chang (Member, IEEE) received the B.S. degree in automation from the University of Electronic Science and Technology of China, Chengdu, China, in 2011, the M.S. degree in pattern recognition and intelligent systems from the Huazhong University of Science and Technology, Wuhan, China, in 2014, and the Ph.D. degree from the School of Artificial Intelligence and Automation, Huazhong University of Science and Technology, in 2019.

From 2014 to 2015, he was a Research Assistant with Peking University, Beijing, China. He was a Research Intern with the Machine Learning Group, Tencent Youtu Lab., Shenzhen, China. He currently holds a post-doctoral position at the AI Center, Peng Cheng Laboratory, Shenzhen. His research interests include multispectral image processing and structural noise removal.

Ting-Zhu Huang received the B.S., M.S., and Ph.D. degrees in computational mathematics from the Department of Mathematics, Xi'an Jiaotong University, Xi'an, China in 1986, 1992, and 2001, respectively.

He is currently a Professor with the School of Mathematical Sciences, University of Electronic Science and Technology of China, Chengdu, China. His current research interests include scientific computation and applications, numerical algorithms for image processing, numerical linear algebra, preconditioning technologies, and matrix analysis with applications.