Rain Streak Removal for Single Image via Kernel Guided CNN

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

Abstract—Rain streak removal is an important issue and has recently been investigated extensively. Existing methods, especially the newly emerged deep learning methods, could remove the rain streaks well in many cases. However the essential factor in the generative procedure of the rain streaks, i.e., the motion blur, which leads to the line pattern appearances, were neglected by the deep learning rain streaks removal approaches and this resulted in over-derain or under-derain results. In this paper, we propose a novel rain streak removal approach using a kernel guided convolutional neural network (KGCNN), achieving the state-of-the-art performance with simple network architectures. We first model the rain streak interference with its motion blur mechanism. Then, our framework starts with learning the motion blur kernel, which is determined by two factors including angle and length, by a plain neural network, denoted as parameter net, from a patch of the texture component. Then, after a dimensionality stretching operation, the learned motion blur kernel is stretched into a degradation map with the same spatial size as the rainy patch. The stretched degradation map together with the texture patch is subsequently input into a derain convolutional network, which is a typical ResNet architecture and trained to output 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 method, which preserves the texture and the contrast while removing the rain streaks.

Index Terms—rain streak removal, guided kernel, convolutional neural network(CNN).

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], event detection [3], action recognition [4], and scene analysis [5]. Therefore, alleviating the effects from the rain is an essential task and has been investigated extensively. Fig. 1 exhibits one example of the single image rain streak removal.

Most of the traditional methods always focus on the discriminative characteristics of the rain streaks and the clean background, for instance, the high-frequency property [8], [9],

* Corresponding authors.

Y.-T. Wang, X.-L. Zhao, T.-X. Jiang, L.-J. Deng and T.-Z. Huang are with the School of Mathematical Sciences, University of Electronic Science and Technology of China, Chengdu, 611731, P. R. China. Y. Chang is with the Institute of Control and Information Technology, Huazhong University of Science and Technology, Wuhan, 430074, P. R. China. E-mails: xlzhao122003@163.com, taixiangjiang@gmail.com, liangjian1987112@126.com, yichang@hust.edu.cn, tingzhuhuang@126.com.
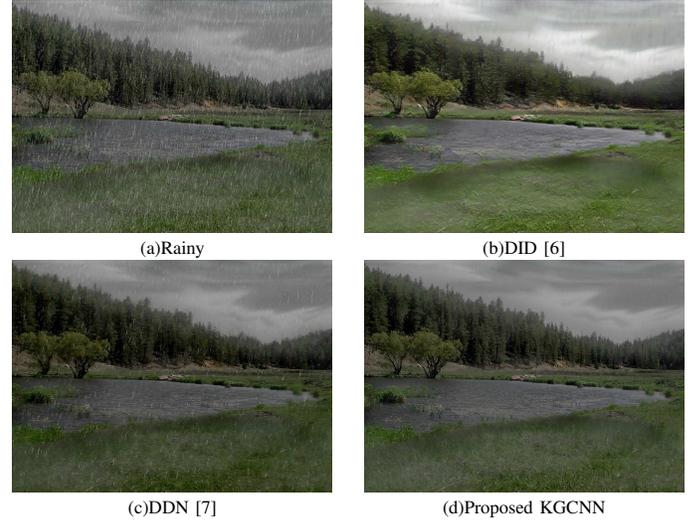

Fig. 1. An example of rain streak removal for a real-world rainy image. **Top-left**: the rainy image. **Top-right**: the derain result by DID [6]. **Button-left**: the derain result by DDN [7]. **Button-right**: the derain result by the proposed KGCNN.

directionality [10]–[14] and repeatability [15] of the rain streaks and the piecewise smoothness [13]–[17] of the background (without loss of generality, we denote the rain-free content as "background" throughout this paper). It is common for the model based methods to elaborately tailor an optimization model with the hand-crafted regularizer expressing the prior knowledge. Although these model based methods go into much depth on the distinct characteristics of the rains streaks, they are often insufficient to cover the majority of the important factors in a real rainy scene since the degradation of rain streaks can be very complex. The traditional learning based methods attempt to overcome this shortage by inferring the discriminative dictionaries [18], the GMMs [16], [17], the stochastic distributions [19] or the convolutional filters [20]–[22] from the data. Benefit from the complex representation ability of the convolutional neural network, the deep learning techniques [6], [20], [23]–[28] further leverage the data to the most extent, and obtain promising results.

As it can be seen in Fig. 1, state-of-the-art de-raining algorithms [6], [7] tend to obtain the under-derain result (bottom-left) or the over-derain result (top-right). We attribute these phenomena to the fact that it is challenging for deraining methods, even the deep learning based methods, to distinguish the rain streaks and the line pattern textures (e.g. the grass in Fig. 1). The rain streaks removal performance of the neural network can be heighten by adopting deeper architecture as [7] or
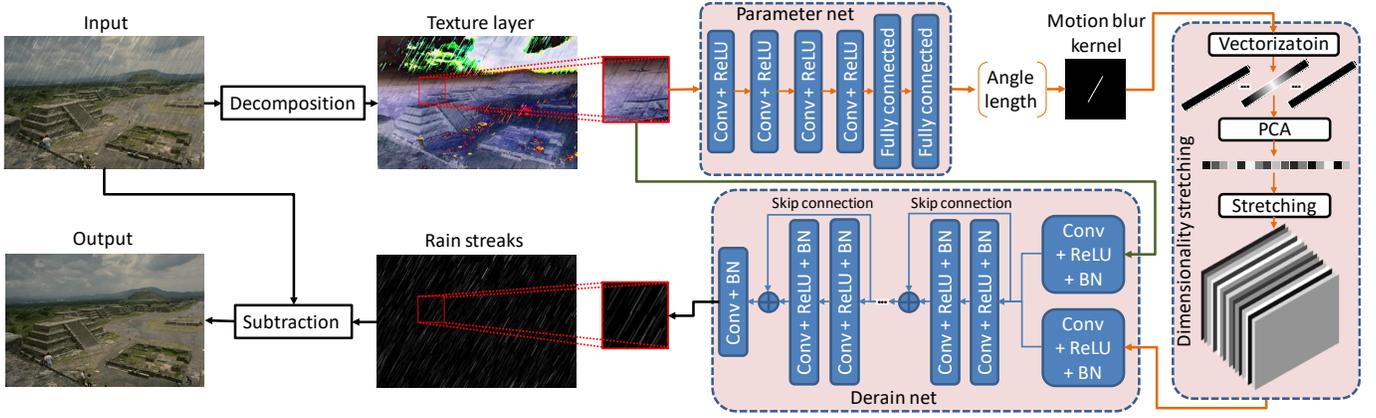
Fig. 2. The flowchart of the proposed KGCNN single image rain streak removal framework. 1) The rainy image is decomposed into the texture component and the structure component. 2) A rainy patch is feed to the parameter net to obtain the angle and the length of the motion blur kernel. 3) The motion blur kernel is stretched to the degradation map. 3) The rainy patch and the degradation map is then transmitted into the derain net, whose output is the rain streak patch. 4) Finally, derain image is obtained by the subtraction of the rainy image and the rain streak image.

elaborately designing the architecture, in which the contexture information are taken into account, as [6], [27]. However, it is still difficult to purposefully enhance the capacity of the neural network to face the fore-mentioned challenge. Meanwhile, anther question is that can we address this challenge and achieve promising performance with the common and simple neural network structures?

In this paper, referring to [1], in which it was pointed out that the appearance of the rain streaks is mainly related to the motion blur mechanism, we propose a novel degradation model of the rain streak interference taking the motion blur procedure into account. By modeling the degradation of rain streaks as the motion blur, we are able to utilize two important distinct characteristics of the rain streaks, i.e., the repeatability and the directionality. The line pattern textures do not possess the same generation mechanism as the rain streaks'. Therefore, this modeling strategy would contribute to distinguish the rain streaks and the line pattern textures. After modeling the rain streaks with motion blur kernel, the questions come to 1) how to infer the motion kernel from the data, and 2) how to utilize the information provided by the motion blur kernel when deraining.

In our approach, we assume that the rain streaks in a small patch approximately share the same motion blur kernel. At the beginning, a rainy patch is feed to the parameter net, a plain 6-layer network, to infer the angle and the length of the motion blur kernel. To enable the learned motion blur kernels to participate in the subsequent deraining process, we adopt the dimensionality stretching strategy [29], which stretched the motion blur kernels to degradation maps with the same spatial resolution as the detail patches. Then, the detail patch together with the degradation map is input into a common 26 layer ResNet, whose output is a patch of rain streaks. Finally, we obtain the derain results by subtracting the rain streaks from rainy image. The core idea of our framework is exploiting the generation mechanism of the rain streaks to guide rain streak removal, and the flowchart of our framework is illustrated in Fig. 2.

Contributions: The contributions of this paper mainly include four aspects.

- We build a novel rain streak generation model which takes the motion blur kernel into account. This modeling strategy enables us to utilize the repeatability and the directionality of the rain streaks.
- A sub-net, i.e., the parameter net, is built to learn the parameters (length and angle) of the motion blur kernel. Unlike existing methods using sub-net to embed the contextual information, our sub-net is designed to exploit the generation information of the rain streaks.
- We propose an effective kernel guided CNN (KGCNN) framework, in which the network structures are common and simple, for rain streak removal. Within this framework, the automatically learned motion blur kernel thoroughly guides the process of rain streak removal.
- Extensive experiments are conducted on publicly available real and synthetic data. Qualitative and quantitative comparisons with existing state-of-the-art methods are presented. The results show that the KGCNN removes rain streaks well while keeping the texture and the contrast of background.

The organization of this paper 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 the synthetic data and the 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 streak interference. [6], [8]–[28], [30]–[43]. Traditionally, these methods can be divided into two categories: single image based methods and multiple-images/videos based methods. Nevertheless, the explosive development of the deep learning brings in a novel branch, i.e., the deep learning methods.

A. Single Image Based Methods

For the single image derain task, Kang *et al.* [8] decomposed a rainy image into low-frequency (LF) and high-frequency (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.* [9] tactfully exploited the structural similarity of the derived HF image bases. Kim *et al.* [44] took advantage of the nonlocal similarity. Chen *et al.* [15] considered the similar and repeated patterns of the rain streaks and the smoothness of the rain-free content. Sparse coding and dictionary learning were adopted by Luo *et al.* [18] and Son *et al.* [45]. In their results, the details of backgrounds were well preserved. The recent work by Li *et al.* [17] utilized the Gaussian mixture model (GMM) patch priors for rain streak 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 [10]–[12], [30] and these methods achieved promising performances. Ren *et al.* [33] removed the rain streaks from the image recovery perspective. Wang *et al.* [32] took advantage of the image decomposition and dictionary learning.

B. Video Based Methods

Garg *et al.* [36] first raised a video rain streak removal method with a comprehensive analysis of the visual effects of rain on an imaging system. Since then, many approaches have been proposed for the video rain streak removal task and obtained good performances with different rain circumstances. Tripathi *et al.* [37] took the spatiotemporal properties into consideration. In [15], the similarity and repeatability of rain streaks were considered, and a generalized low-rank appearance model was proposed. Chen *et al.* [46] considered the highly dynamic scenes. Whereafter, Kim *et al.* [39] considered the temporal correlation of rain streaks and the low-rank nature of clean videos. Santhaseelan *et al.* [40] detected and removed the rain streaks based on phase congruency features. Additionally, comprehensive early existing video based methods were reviewed in [38]. You *et al.* [41] took the raindrops adhered to a windscreen or window glass into account. In [13] and [14], a novel tensor-based video rain streak removal approach was proposed by considering the directional property. The rain streaks and the clean background were stochastically modeled as a mixture of Gaussians by Wei *et al.* [19]. The convolutional sparse coding (CSC), which has shown its ability in image cartoon-texture decomposition [22], was also adopted by Li *et al.* [20] for the video rain streaks removal. Ren *et al.* [42] addressed the video desnow and derain task based on matrix decomposition.

C. Deep Learning Based Methods

The deep learning based method was first applied to derain in [47], in which a 3-layer convolutional neural network (CNN) was designed to remove static raindrops and dirt spots from pictures taken through window glass. Fu *et al.* [34] was the first to successfully tailor a deep CNN for the rain streak removal task. Moreover, in [7], Fu *et al.* designed the deep detail network (DDN) to further improved the performance by adopting the well-known deep residual network (ResNet) [48] structure. Pan *et al.* [49] simultaneously operated on the texture component and the structure component. Yang *et al.* [27] added a binary map, which reflects the contextual information, in the rain streak observation model and constructed a deep network that jointly detected and removed rain streaks. Meanwhile, the increasingly popular Generative Adversarial Networks (GAN) was first used in [35] for rain streak removal and recently applied to the task of dealing with adherent raindrop [23]. In [25], Chen *et al.* proposed a CNN Framework for the video rain streak removal task, while the recurrent neural network was adopted by Liu *et al* [26]. For jointly rain-density estimation and derain, Zhang *et al.* [6] raised a density aware multi-stream densely connected convolutional neural network (DID). In [24], both the rain component and the background component are considered to remove rain streaks. Fan *et al.* [50] developed a residual-guide feature fusion network, which was detachable to meet different rainy conditions. A lightweight pyramid of networks was proposed in [51], using the domain-specific knowledge to simplify the learning process.

III. THE FRAMEWORK OF THE KERNEL GUIDED CONVOLUTIONAL NEURAL NETWORK

In this section, we will give our rain streak observation model and subsequently clarify the detail architecture of the proposed derain framework. As exhibited in Fig. 2, there are mainly three parts, i.e., the parameter net, the dimensionality stretching, and the derain net. The main stream is 1) decomposing the rainy image into texture component and structure component; 2) processing the patches in the texture component using the parameter net, the PCA operation, and the derain net; 3) subtracting the obtained rain streaks and obtaining the derain result.

A. Observation Model

As mentioned previously, the rain streaks can be approximately viewed as sharing the same motion blur kernel. Hence the basic unit of our observation model is the patch. Similar to many existing methods, the rainy image is modeled as a linear superposition:

$$\mathbf{O} = \mathbf{B} + \mathbf{R}, \tag{1}$$

where $\mathbf{O}$, $\mathbf{B}$, and $\mathbf{R} \in \mathbb{R}^{m \times n}$ are patches of the observed rainy image, the underlying background (i.e. the background) and the rain streaks, respectively. After taking the motion blur into consideration, the observation model in Eq. (1) turns to be:

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

where θ and l are respectively the *angle* and *length* of the motion blur kernel $\mathbf{K} \in \mathbb{R}^{p \times p}$, $\mathbf{M}$ is the raindrops mask, and $\otimes$ denotes the convolution operation. Because of the high velocity of the raindrops, the appearance of the rain streaks are mostly linear. Hence using the angle θ and the length l to characterize the motion blur kernel of the rain streaks is reasonable and its advantage illustrated in the next subsection.

Meanwhile, many existing methods conduct the rain streak removal procedures on the detail component [7], [34], [49] or the high-frequency (HF) component [8], [9]. Following this research line, we adopt the guided filter method in [52] as the low-pass filter because it is simple and fast to implement[1]. The rainy patch is decomposed into two parts the texture component $\mathbf{O}_\mathrm{T}$ (denoted as "detail component" in [7], [34], [49]) and the structure component $\mathbf{O}_\mathrm{S}$, and they satisfy $\mathbf{O} = \mathbf{O}_\mathrm{S} + \mathbf{O}_\mathrm{T}$.

The advantages of processing on the texture component have been fully discussed in [7], [34]. In order to facilitate the readers, we briefly bring them herein. It can be found in Fig. 2 that the texture component consists of all rain streaks, i.e., $\mathbf{O}_\mathrm{T} = \mathbf{B}_\mathrm{T} + \mathbf{R}$, so that training and testing on the texture component $\mathbf{O}_\mathrm{T}$ is sufficient and compact. Meanwhile, the texture component is sparser and the range of the values is significantly decreased compared to the pixels in the original image domain. This also decreases the mapping range of the neural network, making the network focus on the important information.

After the decomposition in Eq. (3), the observation model becomes

$$\mathbf{O}_\mathrm{T} = \mathbf{B}_\mathrm{T} + \mathbf{K}(\theta, l) \otimes \mathbf{M}, \tag{3}$$

where $\mathbf{B}_\mathrm{T} \in \mathbb{R}^{m \times n \times c}$ is the rain free content of the texture component and the goal turns to estimate the clean texture part and separate the rain streaks from the rainy texture component. In this work, considering the benefits of processing on the texture component, we attempt to design and train a CNN derainer $\mathcal{F}_\mathrm{D}(\cdot; \Theta_\mathrm{D})$, which maps the texture $\mathbf{O}_\mathrm{T}$ patch into the rain streaks patch $\mathbf{R} = \mathbf{K}(\theta, l) \otimes \mathbf{M}$.

Modeling the rain streaks with the motion blur kernel $\mathbf{K}$ maintains two advantages. One is that two important factors, i.e., the length l and the angle θ, of the rain streak appearance are uniformly encoded by the motion blur kernel. Another one is that the repeatability of the rain streaks allows us to easily infer the two parameters from an input texture patch. In the next subsection, we would present the detail of how to estimate the parameters and embed the learned motion blur kernel to the deraining procedure.

B. The Parameter Sub-Network

Since the CNN has shown its overwhelming superiority on feature extraction, we plan to use a CNN to learn the motion blur kernel. Initially, given a CNN $\mathcal{F}_\mathrm{K}(\cdot; \Theta_\mathrm{K}) : \mathbb{R}^{m \times n} \to \mathbb{R}^{p \times p}$, which maps the input texture patch to the motion blur kernel, with network parameter Θ_K, the loss function for training this CNN architecture is

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

where $\|\cdot\|_F$ denotes the Frobenius norm and i index the patches and motion blur kernels.

However, the performance of $\mathcal{F}_\mathrm{K}(\cdot)$ is not satisfactory. Without the fully connect layer, it dose not converge. 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. This indicates that the intrinsic information lies in a parameter space with much lower dimension than the convolution filter space. Working directly on the low dimension information can not only facilitate the task of motion blur kernel estimation, but also prevent possibly overfitting, which would be verified in the experimental part. Therefore, we adopt the CNN $\mathcal{F}_\mathrm{P}(\cdot; \Theta_\mathrm{P}) : \mathbb{R}^{m \times n} \to \mathbb{R}^2$, which maps the input texture patch to the parameter vector, with network parameter Θ_P and the loss function thereof for training turns to:

$$L_\mathrm{P}(\Theta_\mathrm{P}) = \frac{1}{n} \sum_{i=1}^{n} \|\mathcal{F}_\mathrm{P}(\mathbf{O}_\mathrm{T}^i; \Theta_\mathrm{P})) - \mathbf{p}^i\|_F^2, \tag{5}$$

where $\mathbf{p} = [\theta, l]^\top$ is the parameter vector. The architecture of $\mathcal{F}_\mathrm{P}(\cdot)$ (denoted as "parameter net") is exhibited in Fig. 2. Once the parameters θ and l are determined, the motion blur kernel $\mathbf{K}$ is unique.

C. Dimensionality Stretching

After maintaining the motion blur kernel $\mathbf{K}$, the question comes to how to utilize the motion blur kernel when deraining. In general, the input of the derain net $\mathcal{F}_\mathrm{D}(\cdot; \Theta_\mathrm{D})$, which would be detailed in the next subsection, is supposed to be the texture patch together with the motion blur kernel learned from this texture patch, since the motion blur kernel consists of the prior knowledge of the rain streaks. If we simply splice the texture patch $\mathbf{O}_\mathrm{T} \in \mathbb{R}^{m \times n}$ and motion blur kernel $\mathbf{K} \in \mathbb{R}^{p \times p}$, weight sharing in CNN makes that the texture patch could not get the entire information of the motion blur kernel. Hence, a dimensionality stretching operation in [29] is necessary.

The dimensionality stretching strategy is schematically illustrated in Fig. 2. At the beginning, the motion blur kernel $\mathbf{K}$ 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 degradation maps $\mathcal{M} \in \mathbb{R}^{m \times n \times t}$. All values in the j-th horizontal slice with size $m \times n$ of the 3-dimensional $\mathcal{M}$ are same as the j-th element of $\mathbf{k}_t$. By doing so, the degradation maps then can be concatenated with the texture patch, making CNN possible to handle the two inputs.

However, different from [29], in the case of plain motion blur kernel with only two parameters, we found that t is still too large compared with number of channels of the texture image. Meanwhile, since that working on texture component leads to the pixel values being close to zero, the information of the texture image may be drowned in the information of motion blur kernel with a relatively large t. To tackle this issue, degradation maps will be concatenated with the texture image after the first convolutional layer in the derain net, as shown in Fig. 2.

D. Derain Net

As previously mentioned in Sec I, instead of elaborately designing the architecture, we resort to the typical ResNet structure. A cascade of 3×3 convolutional layers are applied to

---

[1]As discussed in [34], the choice of low-pass filter is not limited to guided filtering.

perform the deraining. Each layer is composed of three types of operations, including convolution (denoted as "Conv"), rectified linear units [53] (denoted as "ReLU"), and batch normalization [54] (denoted as "BN"). We still use Frobenius norm in the loss function, which is

$$L_{\mathrm{D}}(\Theta_{\mathrm{D}}) = \frac{1}{n} \sum_{i=1}^{n} \|f_d(\mathbf{O}^i_{texture}, \mathbf{K}^i - \mathbf{R}^i)\|_F^2, \qquad (6)$$

where $\mathbf{R}$ is rain streaks. After subtracting the rain streaks $\mathbf{R}$ from the rainy image $\mathbf{O}$, we could get the background.

Disscusion: As we mentioned above, 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 streak 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, are automatically learned. The embedding of the motion blur kernel into the derain net, which maintains a plain ResNet structure, greatly enhances the performance (see the comparisons in Sec. IV-E). To some extent, the utilization of the motion blur kernel in our method can be viewed as the traditional optimization model utilizing the regularizer to express the prior knowledge.

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 our KGCNN with six state-of-the-art methods, including three traditional methods: the unidirectional global sparse model (UGSM[2]) [11], the discriminative sparse coding method (DSC[3]) [18], and the method using layer prior (LP[4]) [16], as well as three deep learning based methods: the density-aware multi-stream deraining dense network (DID[5]) [6], a plain convolutional neural network deraining method (CNN[6]) [34], and the deep detail network (DDN[7]) [7].

A. Rainy Images Simulation

With the observation model in Eq. (2), the synthetic rainy images are generated by the following steps. (1) Transform the background from RGB color space to YUV color space[8]. (2) Generate the raindrops mask $\mathbf{M}$ by adding salt and pepper noise with signal-noise ratio from 0.9 to 1.0 to a zero matrix with the same size as the Y channel of the background, and adding a Gaussian blur with standard variance from 0.2 to 0.5. (3) Generate the motion blur kernel $\mathbf{K}$ with angle θ sampled from $[45°, 135°]$ and length l varing from 15 to 30. (5) Directly added the generated rain streaks $\mathbf{R} = \mathbf{K} \otimes \mathbf{M}$ to the background on Y channel, and the intensity values greater than 1 are set as 1. (6) Finally, transform the image back to RGB color space.

---

[2]http://www.escience.cn/people/dengliangjian/index.html

[3]http://www.math.nus.edu.sg/~matjh/research/research.htm

[4]http://yu-li.github.io/

[5]https://github.com/hezhangsprinter/DID-MDN

[6]https://xueyangfu.github.io/projects/tip2017.html

[7]https://xueyangfu.github.io/projects/cvpr2017.html

[8]https://en.wikipedia.org/wiki/YUV

B. Experiments Setting

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 datasets do not consist of the information of the motion blur kernel, we train our networks only on our synthetic data. The patch size is set as $64 \times 64 \times 3$. Guided filter with radius 15 and regularization 1 is selected to decompose the rain images. By preserving 99% of the energy, the kernel is projected onto a space of dimension 162. Because of the full connection, the input image should be split into several patches for experiments. We use Adam [55] optimizers with 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 6GB. For other compared methods based in Matlab, they are running on Matlab 2017A.

C. Synthesized Data

In this subsection, we evaluate performance of different state-of-the-art methods on the synthetic rainy images. Three datasets are selected: 1) the benchmark dataset provided by Dr. Yu Li using the rain streaks rendering technique in [56] (denoted as Rain12), 2) 3 synthetic rainy images by our simulating method, and 3) several synthetic rainy images in [11]. Due to the limit of space, we only show partial results in this section, and please see more results in the supplementary materials.

For quantitative comparisons, we adopt the peak signal to noise ratio (PSNR), structure similarity index (SSIM) [57], feature similarity index (FSIM) [58], universal image quality index (UIQI) [59], and gradient magnitude similarity deviation (GMSD, smaller is better) [60] as the quality metrics of the deraining results. Particularly, since the compared methods are implemented with different programming languages (or platforms), e.g., UGSM with Matlab and CNN with Python, we save all output images of different methods as png format, then reload them in Matlab and compute the corresponding quantitative results on RGB color space.

To show that KGCNN could remove rain streaks while keeping the texture and the contrast of background, we show the rain streak images (residual images between rainy images and resulted images).

Normalization is performed to the rain streak images so that we could distinguish whether the proposed method changes texture and contrast significantly or removes rain streaks completely. For instance, if the rain streak images are too bright, it indicates the method significantly changes intensity contrast. For the first dataset, Fig. 3 shows the visual results, local close-up images and rain streak images on one synthetic rainy image of the Rain12 dataset. We can see that the proposed KGCNN method could remove the rain streaks completely while other approaches fail to do so (see local close-up images for better comparisons in Fig. 3). Especially, it is easy to see that the obtained rain streaks by the proposed approach do not contain the structures of background, which indicates KGCNN has a very good ability for rain streak removal. From the perspective of quantitative results, KGCNN method performs best for the
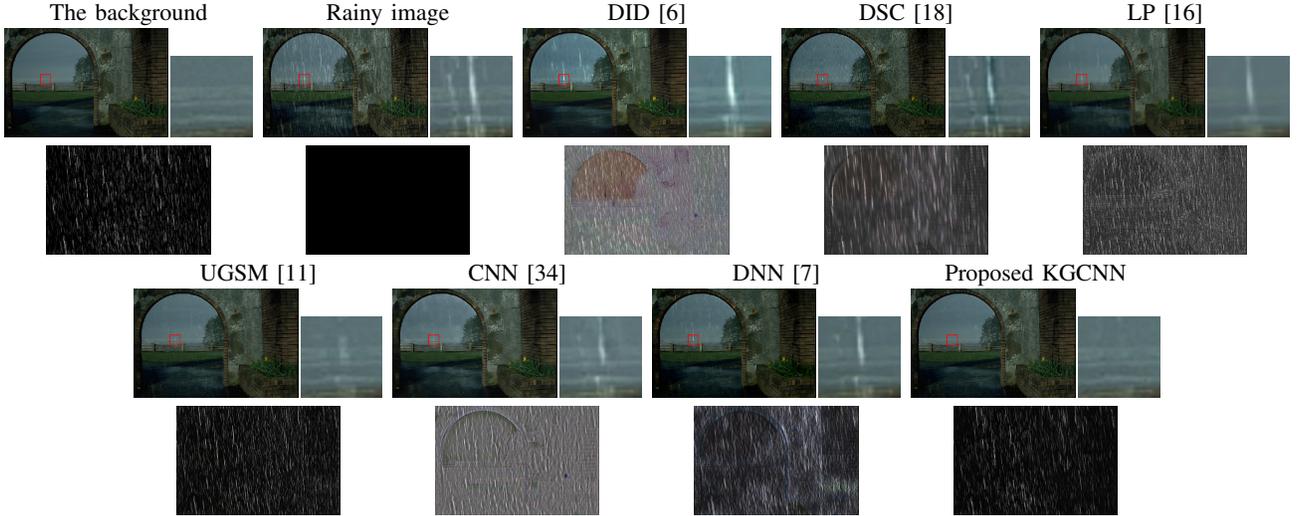
Fig. 3. Rain streak removal results by different methods on one rainy image of the Rain12 dataset.

12 synthesized images, compared with other six state-of-the-art methods (see Table I for more details).

TABLE I
QUANTITATIVE COMPARISONS OF RAIN STREAK REMOVAL RESULTS BY DID [6], DSC [18], LP [61], UGSM [62], CNN [34], DDN [7], AND KGCNN ON THE RAIN12 (AVERAGE VALUE).

Method	PSNR	SSIM	FSIM	UIQI	GMSD	Time (s)
rainy	28.822	0.910	0.910	0.968	0.134	-
DID	27.485	0.919	0.918	0.941	0.086	**0.624**
DSC	28.584	0.915	0.917	0.965	0.097	153.792
LP	30.825	0.947	0.935	0.966	0.070	321.328
UGSM	32.185	0.958	0.947	**0.983**	0.065	2.767
CNN	28.155	0.942	0.935	0.966	0.071	7.432
DNN	29.112	0.935	0.942	0.931	0.073	0.660
KGCNN	**34.731**	**0.971**	**0.965**	**0.983**	**0.055**	8.850

For the second dataset, we generate another 3 synthesized rainy images (road, night, and street) for test. Some of resulted derain images by different methods are selected to be shown in Fig. 4 and Fig. 5. The visual results also demonstrate that the KGCNN method not only removes rain streaks completely, but also preserves the background information well. We report the quantitative performance of the derain results obtained by different approaches in Table II, which shows the superiority of the KGCNN method.

UGSM performs quite competitively for Rain12.Therefore, it is necessary to take more test images from UGSM (tree, panda, and bamboo) to compare the rain removal ability of the proposed method and UGSM method. In addition, based on the code provided by the authors in [62], we also change the rain streaks' angles of these images from UGSM to generate three new synthesize images (tree2, panda2, and bamboo2) for testing. Fig. 6 and Fig. 7 respectively present the visual and rain streak results of these images from [62], which indicates the superiority of the proposed method.

In Fig. 8 and Fig. 9, rain streaks with very large angles are added to background to formulate rainy images. Noting that the UGSM is based on directional priors, which is quite effective to the case of vertical rain streaks, but less effective

TABLE II
QUANTITATIVE COMPARISONS OF RAIN STREAK REMOVAL RESULTS BY DID [6], DSC [18], LP [61], UGSM [62], CNN [34], DDN [7], AND KGCNN ON 3 SYNTHETIC RAINY IMAGES BY OUR SIMULATING METHOD.

Images	Method	PSNR	SSIM	FSIM	UIQI	GMSD	Time (s)
	rainy	18.171	0.809	0.909	0.824	0.150	-
	DID	22.409	0.889	0.927	1.005	0.104	**0.625**
	DSC	22.302	0.863	0.926	**1.164**	0.122	197.776
	LP	22.227	0.889	0.919	0.911	0.122	298.329
road	UGSM	22.763	0.909	0.933	0.911	0.105	11.703
	TIP	18.045	0.836	0.911	0.830	0.124	12.419
	CVPR	23.325	0.902	0.921	0.875	0.117	0.843
	KGCNN	**27.851**	**0.957**	**0.957**	0.978	**0.076**	9.023
	rainy	18.506	0.539	0.787	0.753	0.261	-
	DID	25.294	0.851	0.924	0.813	0.099	**0.625**
	DSC	22.219	0.598	0.835	0.904	0.208	200.172
	LP	23.034	0.780	0.872	0.806	0.175	297.194
night	UGSM	20.626	0.641	0.824	0.784	0.225	5.568
	TIP	19.215	0.648	0.851	0.757	0.193	9.185
	CVPR	23.781	0.674	0.882	0.846	0.172	0.375
	KGCNN	**31.231**	**0.957**	**0.970**	**0.930**	**0.046**	8.848
	rainy	18.798	0.837	0.882	0.968	0.162	-
	DID	22.359	0.865	0.902	0.975	0.128	**0.640**
	DSC	21.519	0.846	0.898	0.992	0.141	307.007
	LP	22.393	0.894	0.907	0.987	0.145	276.631
tree	UGSM	22.382	0.913	0.921	0.985	0.111	8.236
	TIP	18.793	0.872	0.903	0.969	0.126	14.075
	CVPR	24.013	0.900	0.908	0.994	0.121	0.845
	KGCNN	**28.031**	**0.966**	**0.954**	**0.997**	**0.073**	8.751

for the case of oblique rain streaks. Therefore, from Fig. 8 and Fig. 9, we can know that the proposed KGCNN method performs significantly better than UGSM method, both visually and quantitatively. Moreover, the KGCNN method also exhibits better ability of rain streak removal, compared with other state-of-the-art methods. Table III also demonstrate the effectiveness of our method from the perspective of quantitative results.

D. Real-world data

For real-world data, since the ground truth images are unknown, we do not give the quantitative comparisons and only evaluate the performance of different methods visually, including the derain images and the rain streak images. From Fig. 10, the methods of DID, DDN, and KGCNN exhibit
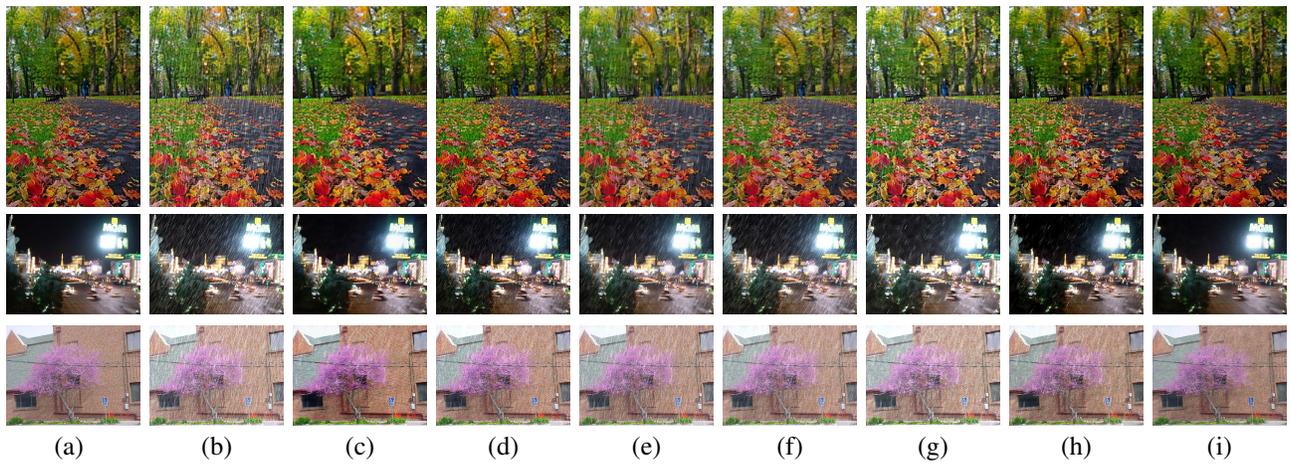
Fig. 4. Rain streak removal results by different methods on 3 synthetic rain images (road, night, and street) by our simulating method. From left to right: (a) the background, (b) the rainy images, the derain results by (c) DID [6], (d) DSC [18], (e) LP [61], (f) UGSM [62], (g) CNN [34], (h) DDN [7], and (i) KGCNN.
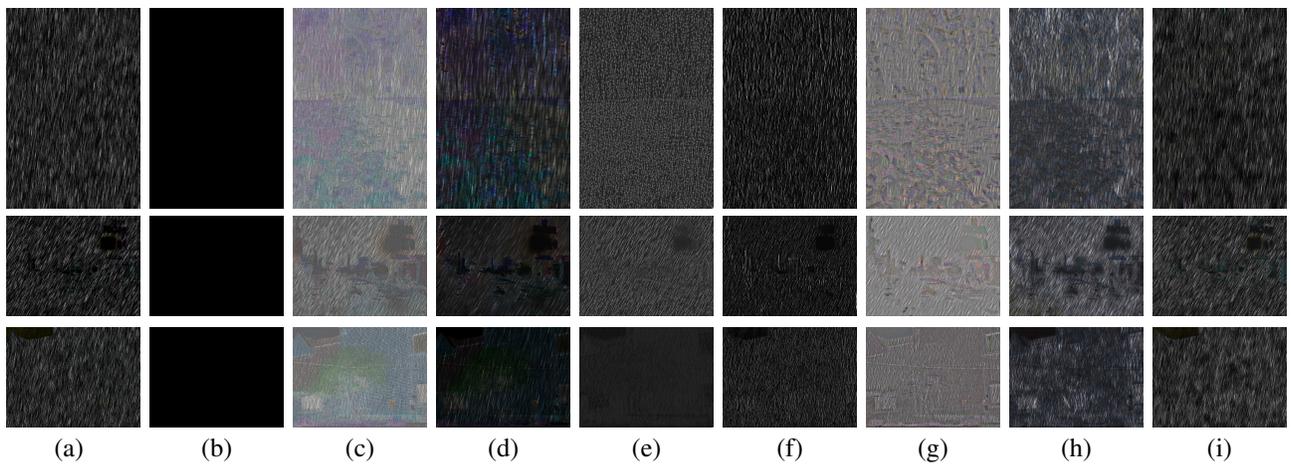

Fig. 5. The rain streak images of the rain streak removal results by different methods on 3 synthetic rain images (road, night, and street) by our simulating method. From left to right: (a) the background, (b) the rainy images, the derain results by (c) DID [6], (d) DSC [18], (e) LP [61], (f) UGSM [62], (g) CNN [34], (h) DDN [7], and (i) KGCNN.
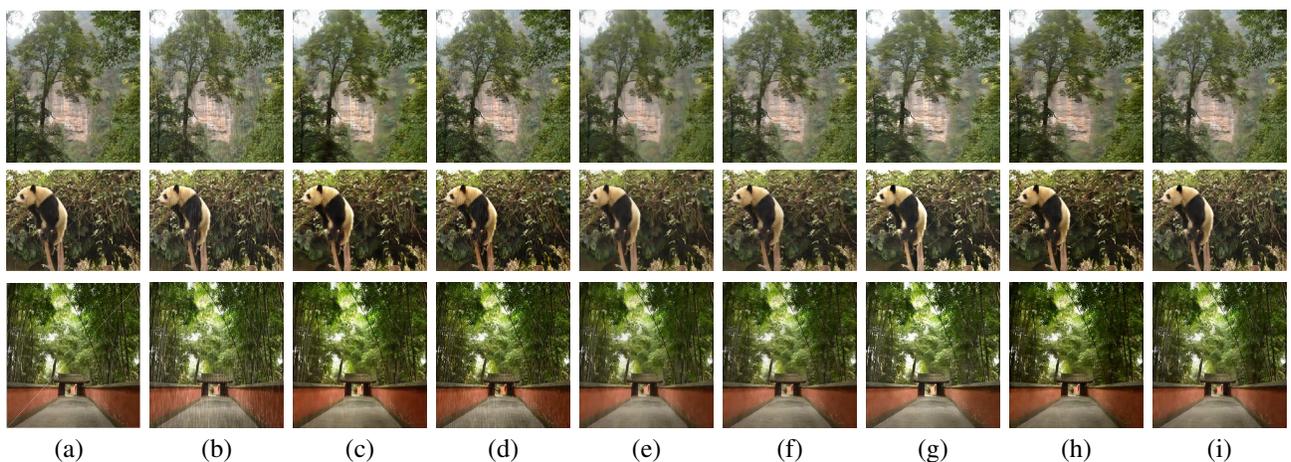

Fig. 6. Rain streak removal results by different methods on 3 synthetic rainy images (tree, panda, and bamboo) selected from [62]. From left to right: (a) the background, (b) the rainy images, the derain results by (c) DID [6], (d) DSC [18], (e) LP [61], (f) UGSM [62], (g) CNN [34], (h) DDN [7], and (i) KGCNN.

similar visual results and remove rain streaks completely, while other approaches fail to remove all rain streaks. In addition, from the rain streak images in Fig. 10, the methods of DID and DDN fail to separate the rain streaks and background
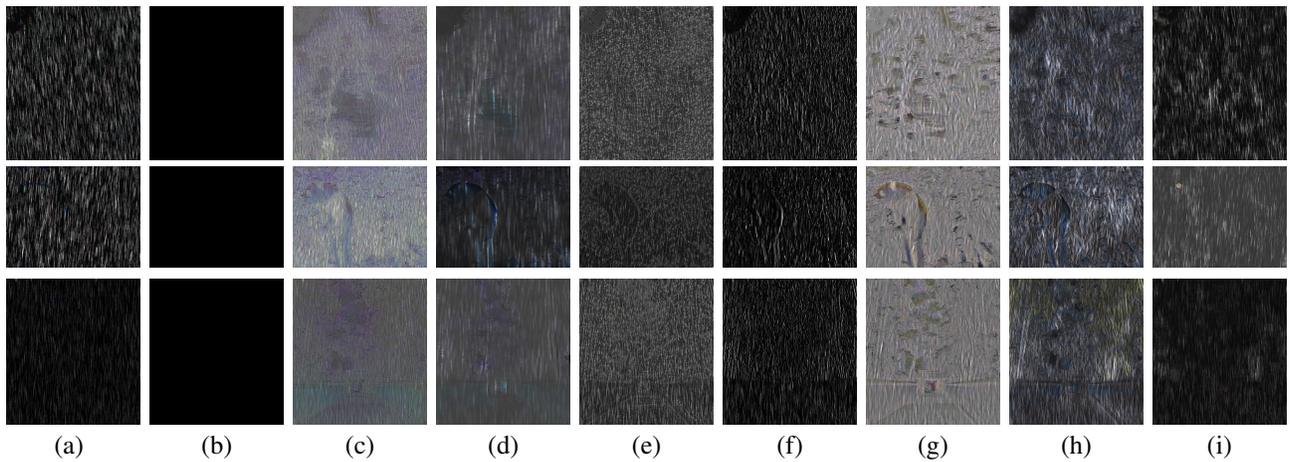
Fig. 7. The rain streak images of the rain streak removal results by different methods on 3 synthetic rainy images (tree, panda, and bamboo) selected from [62]. From left to right: (a) the background, (b) the rainy images, the derain results by (c) DID [6], (d) DSC [18], (e) LP [61], (f) UGSM [62], (g) CNN [34], (h) DDN [7], and (i) KGCNN.
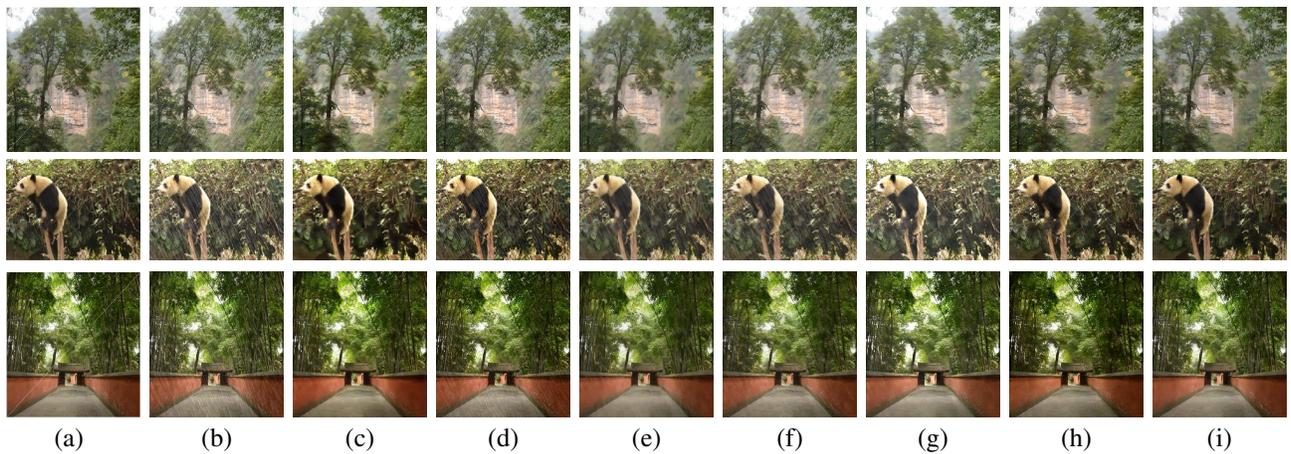

Fig. 8. Rain streak removal results by different methods on 3 new synthetic rainy images (tree2, panda2, and bamboo2). From left to right: (a) the background, (b) the rainy images, the derain results by (c) DID [6], (d) DSC [18], (e) LP [61], (f) UGSM [62], (g) CNN [34], (h) DDN [7], and (i) KGCNN.
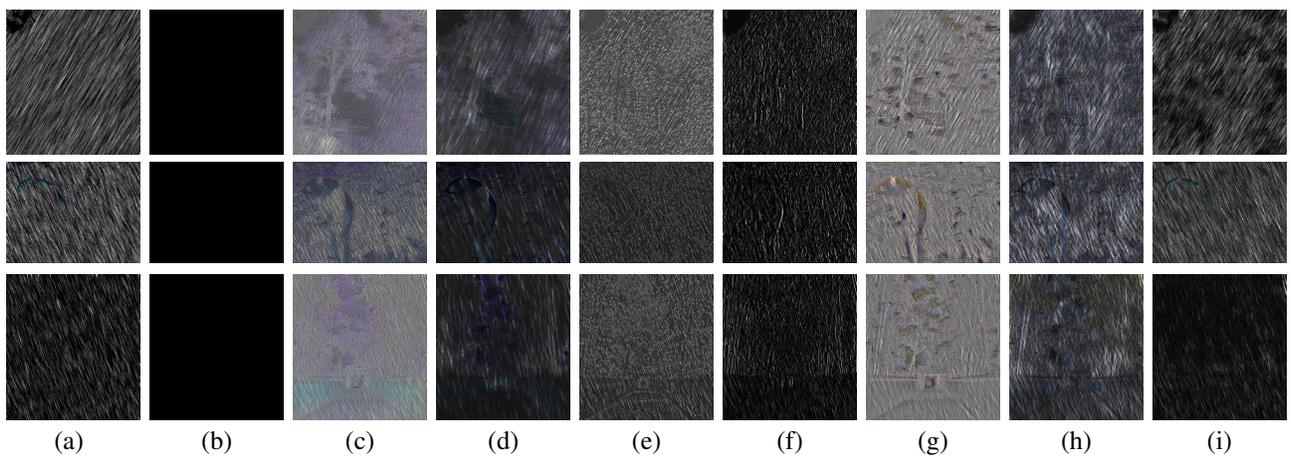

Fig. 9. The rain streak images of the rain streak removal results by different methods on 3 new synthetic rainy images (tree2, panda2, and bamboo2). From left to right: (a) the background, (b) the rainy images, the derain results by (c) DID [6], (d) DSC [18], (e) LP [61], (f) UGSM [62], (g) CNN [34], (h) DDN [7], and (i) KGCNN.

texture well and leave some background texture to rain streaks, which demonstrates that the networks of DID and DDN could not distinguish background texture and the rain streaks well. Moreover, the DID method also changes image contrast

TABLE III
QUANTITATIVE COMPARISONS OF RAIN STREAK REMOVAL RESULTS BY DID [6], DSC [18], LP [61], UGSM [62], CNN [34], DDN [7], AND KGCNN ON
3 SYNTHETIC RAINY IMAGES SELECTED IN [62].

Images	Rain type Method	original images						images with large angle					
		PSNR	SSIM	FSIM	UIQI	GMSD	Time (s)	PSNR	SSIM	FSIM	UIQI	GMSD	Time (s)
tree	rainy	27.375	0.934	0.942	0.989	0.076	-	21.475	0.895	0.926	0.952	0.078	-
	DID	27.826	0.926	0.937	0.982	0.060	**0.625**	25.076	0.916	0.929	0.989	0.069	**0.615**
	DSC	29.646	0.940	0.944	0.996	0.064	87.726	24.466	0.908	0.924	0.982	0.073	89.726
	LP	29.435	0.927	0.912	0.997	0.076	226.091	25.497	0.909	0.899	0.983	0.081	236.091
	UGSM	30.659	0.954	0.948	0.998	0.056	0.979	22.842	0.906	0.927	0.964	0.074	0.979
	CNN	26.532	0.942	0.941	0.989	0.065	6.139	21.201	0.915	0.934	0.952	0.068	6.339
	DDN	26.944	0.944	0.945	0.977	0.066	0.812	28.640	0.936	0.935	**0.996**	0.069	0.812
	KGCNN	**32.269**	**0.971**	**0.966**	**0.998**	**0.049**	9.402	**29.393**	**0.960**	**0.956**	0.993	**0.057**	10.422
panda	rainy	27.102	0.920	0.946	0.978	0.083	-	17.569	0.750	0.871	0.836	0.140	-
	DID	27.120	0.923	0.946	0.952	0.066	0.625	24.077	0.891	0.914	0.960	0.094	0.665
	DSC	26.568	0.914	0.940	0.968	0.077	70.239	21.688	0.790	0.867	0.959	0.124	70.639
	LP	29.250	0.938	0.943	0.994	0.073	133.709	21.197	0.857	0.904	0.908	0.105	143.709
	UGSM	27.823	0.925	0.926	0.994	0.082	2.505	19.621	0.798	0.876	0.885	0.118	2.750
	CNN	24.838	0.927	0.937	0.976	0.070	3.421	17.145	0.791	0.883	0.832	0.105	4.442
	DDN	25.693	0.921	0.947	0.934	0.077	**0.562**	23.332	0.868	0.904	0.972	0.099	**0.562**
	KGCNN	**30.958**	**0.964**	**0.967**	**0.994**	**0.055**	6.997	**27.130**	**0.925**	**0.937**	**0.977**	**0.089**	7.907
bamboo	rainy	26.091	0.923	0.930	0.966	0.102	-	26.997	0.944	0.938	0.967	0.090	-
	DID	27.355	0.939	0.930	0.964	0.071	0.625	27.987	0.949	0.938	0.990	0.062	0.635
	DSC	26.426	0.923	0.925	0.964	0.086	120.959	27.374	0.942	0.933	0.959	0.080	124.459
	LP	29.337	0.954	0.933	0.953	0.071	218.103	29.594	0.960	0.936	1.032	0.069	213.103
	UGSM	27.647	0.936	0.919	**0.991**	0.086	2.794	27.514	0.933	0.916	0.990	0.088	2.679
	CNN	25.761	0.941	0.926	0.975	0.081	5.499	25.877	0.946	0.932	0.966	0.075	5.499
	DDN	24.385	0.905	0.926	0.843	0.097	**0.531**	25.429	0.924	0.935	0.898	0.087	**0.521**
	KGCNN	**29.587**	**0.963**	**0.955**	0.962	**0.070**	9.058	**31.303**	**0.976**	**0.965**	1.113	**0.056**	9.445

significantly.

In Fig. 11, the KGCNN method removes the rain streaks completely while other approaches still exist obvious rain streaks. From the rain streak images, our method could separate rain streaks excellently, while other method leaves some background texture to the separated rain streaks. Especially the visual result by DID method shows a little of blur effect due to the loss of the texture information. Particularly, readers can find more results in Fig. 12 which also verifies the superiority of the proposed KGCNN method.

E. Influence of kernel in the KGCNN method

In this paper, we propose a kernel guided CNN method for the image rain streak removal application. The kernel plays a very important role to the KGCNN method. There are still two problems. (1) Dose the derain net output the rain streaks only using the information of rainy image and ignoring the kernel information? (2) Dose the derain net work better if we retain it without the input of kernel? To show the influence of the kernel in our method, we discard the kernel guided assumption to see the results what will happen with Rain12. We use KGCNN to represent the proposed method, $KGCNN^a$ to represent the proposed method with kernel information being zero, and $KGCNN^b$ to represent the detrain net trained individually with our training data. Fig. 13 shows the visual results there methods. It is easy to know that the kernel plays an import role in KGCNN. The derain net dose use the kernel to output the rain streaks (see the result of $KGCNNa$) and

even we train the derain net individually, the result is still good enough (see the result of $KGCNN^b$). The quantitative results in Table IV also demonstrate the similar conclusion. In summary, the kernel guided assumption is quite important to the framework of the KGCNN method.

TABLE IV
QUANTITATIVE COMPARISONS OF RAIN STREAK REMOVAL RESULTS BY
$KGCNN^a$, $KGCNN^b$, AND KGCNN ON RAIN12 (AVERAGE VALUE).

Method	PSNR	SSIM	FSIM	UIQI	GMSD
rainy	28.822	0.910	0.910	0.968	0.134
$KGCNN^a$	29.855	0.924	0.919	0.967	0.127
$KGCNN^b$	33.194	0.960	0.950	**0.986**	0.065
KGCNN	**34.731**	**0.971**	**0.965**	0.983	**0.055**

F. Discussions on the depth and breadth

Increasing the depth of network or increasing the filter number of network can improve a network's capacity. We also investigate the optimal network design to achieve the best derain results. In this section, we test the impact of network depth and width of KGCNN on Rain12. Especially, we test for depth $\in \{18, 26, 34\}$ and filter number $\in \{24, 36, 48\}$. Table V shows the average values of the quantitative results. As is clear, adding more hidden layers achieves better results over increasing the number of filters per layer while increasing computational time. But we could see that there is over-fitting
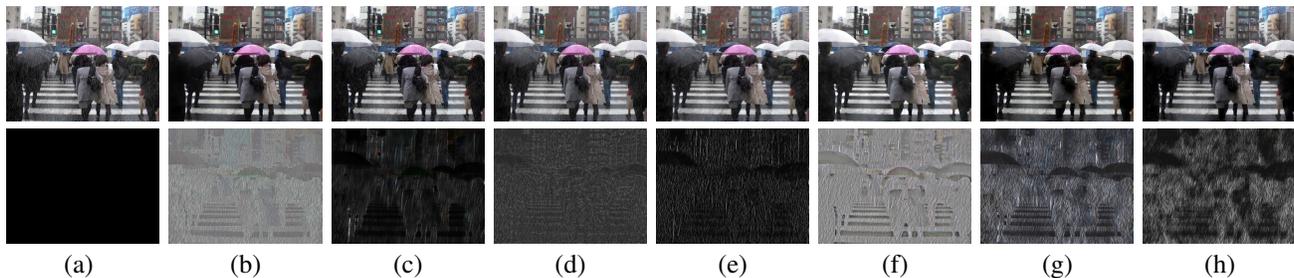
Fig. 10. Rain streak removal results and rain streak images by different methods on real rainy images. From left to right: (a) the rainy images, the results by (b) DID [6], (c) DSC [18], (d) LP [61], (e) UGSM [62], (f) CNN [34], (g) DDN [7], and (h) KGCNN.
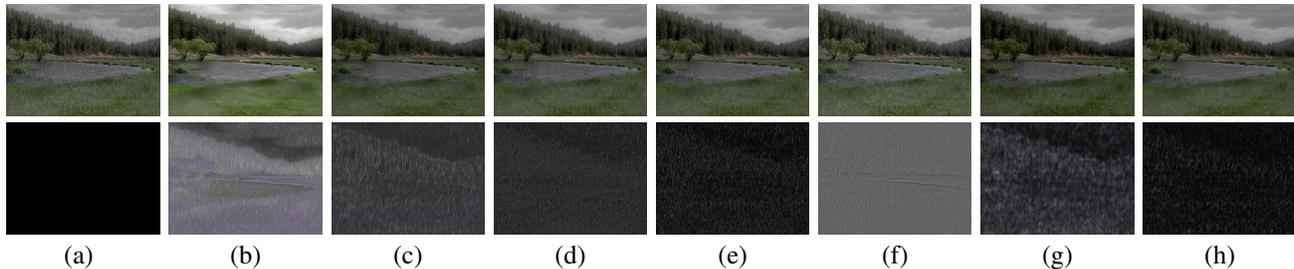

Fig. 11. Rain streak removal results and rain streak images by different methods on real rainy images. From left to right: (a) the rainy images, the results by (b) DID [6], (c) DSC [18], (d) LP [61], (e) UGSM [62], (f) CNN [34], (g) DDN [7], and (h) KGCNN.

when depth is 34 and filter numbers is 48. To balance the performance between avoiding the over-fitting and reducing the computation, we choose 26 as depth and 36 as filter number for our experiments above.

TABLE V
QUANTITATIVE COMPARISONS OF RAIN STREAK REMOVAL RESULTS BY DIFFERENT DEPTH AND FILTER NUMBER ON RAIN12(AVERAGE VALUE).

depth	filter number	PSNR	SSIM	FSIM	UIQI	GMSD
18	24	34.049	0.969	0.963	**0.991**	0.057
	36	34.718	**0.973**	0.966	0.985	0.053
	48	34.738	0.972	**0.967**	0.979	0.052
26	24	34.614	0.972	0.965	0.987	0.053
	36	34.731	0.971	0.965	0.983	0.055
	48	**34.941**	0.972	0.966	0.991	0.053
34	24	34.794	0.972	0.966	0.988	**0.051**
	36	34.853	0.972	0.966	0.986	0.051
	48	34.414	0.969	0.963	0.983	0.059

V. CONCLUSION

We have presented a deep learning architecture called KGCNN for removing rain streaks for single images. Using guided kernel on the texture component, our approach learns the mapping function between rain image on detail component and rain streaks. We show that convolutional neural networks, a technology widely used for high-level vision task, with guided kernel can also be exploited to successfully deal with natural images under bad weather conditions. We also show that KGCNN noticeably outperforms other state-of-the-art methods with respect to image quality. In addition, by using guided kernel, we are able to show that we do not need a very complex network to perform rain streak removal.

REFERENCES

[1] K. Garg and S.-K Nayar. Vision and rain. *International Journal of Computer Vision*, 75(1):3–27, 2007.

[2] X. Zhang, C. Zhu, S. Wang, Y.-P. Liu, and M. Ye. A bayesian approach to camouflaged moving object detection. *IEEE Transactions on Circuits and Systems for Video Technology*, 27(9):2001–2013, 2017.

[3] M.-S Shehata, J. Cai, W.-M. Badawy, T.-W Burr, M.-S Pervez, R.-J Johannesson, and A. Radmanesh. Video-based automatic incident detection for smart roads: the outdoor environmental challenges regarding false alarms. *IEEE Transactions on Intelligent Transportation Systems*, 9(2):349–360, 2008.

[4] S.-J. Song, C.-L. Lan, J.-L. Xing, W.-K. Zeng, and J.-Y. Liu. Spatio-temporal attention-based lstm networks for 3d action recognition and detection. *IEEE Transactions on Image Processing*, 27(7):3459–3471, 2018.

[5] L. Itti, C. Koch, E. Niebur, et al. A model of saliency-based visual attention for rapid scene analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(11):1254–1259, 1998.

[6] H. Zhang and V.-M Patel. Density-aware single image de-raining using a multi-stream dense network. *arXiv preprint arXiv:1802.07412*, 2018.

[7] X.-Y. Fu, J.-B. Huang, D.-L. Zeng, Y. Huang, X.-H. Ding, and J. Paisley. Removing rain from single images via a deep detail network. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1715–1723, 2017.

[8] L.-W. Kang, C.-W. Lin, and Y.-H. Fu. Automatic single-image-based rain streaks removal via image decomposition. *IEEE Transactions on Image Processing*, 21(4):1742–1755, 2012.

[9] S.-H. Sun, S.-P. Fan, and Y.-C. Wang. Exploiting image structural similarity for single image rain removal. In *the IEEE International Conference on Image Processing (ICIP)*, pages 4482–4486, 2014.

[10] Y. Chang, L.-X. Yan, and S. Zhong. Transformed low-rank model for line pattern noise removal. In *the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1726–1734, 2017.
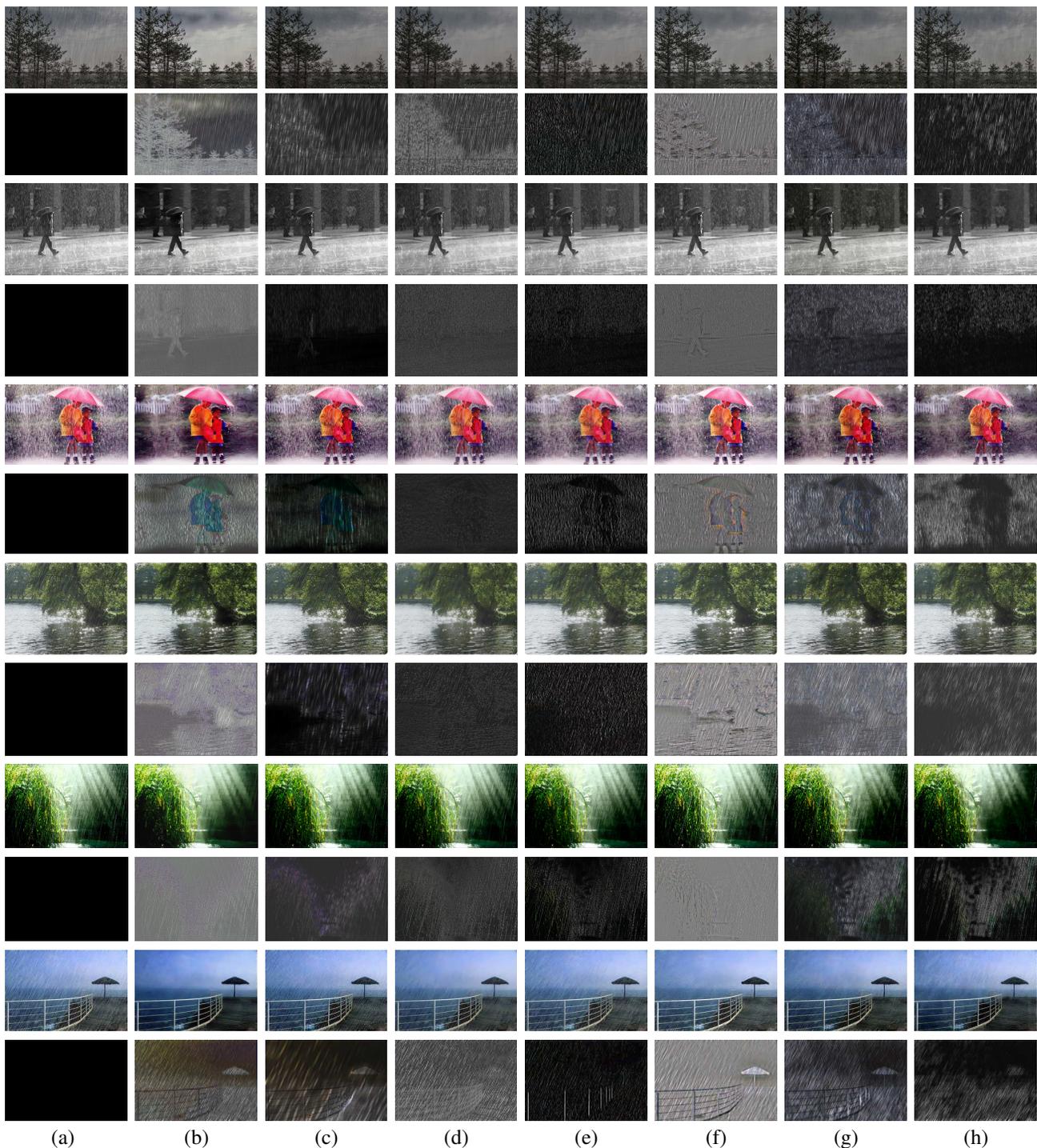
Fig. 12. Rain streak removal results and rain streak images by different methods on real rainy images. From left to right: (a) the rainy images, the results by (b) DID [6], (c) DSC [18], (d) LP [61], (e) UGSM [62], (f) CNN [34], (g) DDN [7], and (h) KGCNN.

[11] L.-J. Deng, T.-Z. Huang, X.-L. Zhao, and T.-X. Jiang. A directional global sparse model for single image rain removal. *Applied Mathematical Modelling*, 59:662–679, 2018.

[12] S.-L. Du, Y.-G. Liu, M. Ye, Z.-Y. Xu, J. Li, and J.-G. Liu. Single image deraining via decorrelating the rain streaks and background scene in gradient domain. *Pattern Recognition*, 79:303–317, 2018.

[13] 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 *the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4057–4066, July 2017.

[14] 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. *arXiv preprint arXiv:1803.07487*, 2018.

[15] Y.-L. Chen and C.-T. Hsu. A generalized low-rank appearance model for spatio-temporally correlated rain streaks. In *the IEEE International Conference on Computer Vision (ICCV)*, pages 1968–1975, 2013.

[16] Y. Li, R.-T Tan, X.-J. Guo, J.-B. Lu, and M.-S Brown. Rain streak removal using layer priors. In *the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2736–2744, 2016.

[17] Y. Li, R.-T Tan, X.-J. Guo, J.-B. Lu, and M.-S Brown. Single image rain streak decomposition using layer priors. *IEEE Transactions on Image Processing*, 26(8):3874–3885, 2017.
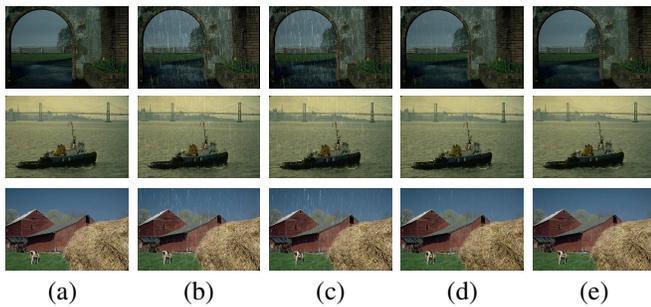
Fig. 13. Rain streak removal results by different methods on 3 selected images from Rain12 dataset. From left to right: (a) the background, (b) the rainy images, the derain results by (c) KGCNN^a, (d) KGCNN^b, and (d) KGCNN.

[18] Y. Luo, Y. Xu, and H. Ji. Removing rain from a single image via discriminative sparse coding. In *the IEEE International Conference on Computer Vision (ICCV)*, pages 3397–3405, 2015.

[19] W. Wei, L.-X. Yi, Q. Xie, Q. Zhao, D.-Y. Meng, and Z.-B. Xu. Should we encode rain streaks in video as deterministic or stochastic? In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2516–2525, 2017.

[20] M.-H. Li, Q. Xie, Q. Zhao, W. Wei, S.-H. Gu, J. Tao, and D.-Y. Meng. Video rain streak removal by multiscale convolutional sparse coding. In *the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[21] S.-H. Gu, D.-Y. Meng, W.-M. Zuo, and L. Zhang. Joint convolutional analysis and synthesis sparse representation for single image layer separation. In *the IEEE International Conference on Computer Vision (ICCV)*, pages 1717–1725. IEEE, 2017.

[22] H. Zhang and V.-M Patel. Convolutional sparse and low-rank coding-based rain streak removal. In *the IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 1259–1267. IEEE, 2017.

[23] R. Qian, Robby T. Tan, W.-H. Yang, J.-J. Su, and J.-Y. Liu. Attentive generative adversarial network for raindrop removal from a single image. In *the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[24] S.-Y. Li, W.-Q. Ren, J.-W. Zhang, J. Yu, and X.-J. Guo. Fast single image rain removal via a deep decomposition-composition network. *arXiv preprint arXiv:1804.02688*, 2018.

[25] J. Chen, C.-H. Tan, J.-H. Hou, L.-P. Chau, and H. Li. Robust video content alignment and compensation for rain removal in a cnn framework. *arXiv preprint arXiv:1803.10433*, 2018.

[26] J.-Y. Liu, W.-H. Yang, S. Yang, and Z.-M. Guo. Erase or fill? deep joint recurrent rain removal and reconstruction in videos. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[27] W.-H. Yang, R.-T. Tan, J.-S. Feng, J.-Y. Liu, Z.-M. Guo, and S.-C. Yan. Deep joint rain detection and removal from a single image. In *the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

[28] W. Wei, D.-Y. Meng, Q. Zhao, and Z.-B. Xu. Semi-supervised cnn for single image rain removal. *arXiv preprint arXiv:1807.11078*, 2018.

[29] K. Zhang, W.-M Zuo, and L. Zhang. Learning a single convolutional super-resolution network for multiple degradations. *arXiv preprint arXiv:1712.06116*, 2017.

[30] L. Zhu, C.-W. Fu, D. Lischinski, and P.-A. Heng. Joint bi-layer optimization for single-image rain streak removal. In *the IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.

[31] B.-H. Chen, S.-C. Huang, and S.-Y. Kuo. Error-optimized sparse representation for single image rain removal. *IEEE Transactions on Industrial Electronics*, 64(8):6573–6581, 2017.

[32] Y.-L. Wang, S.-C. Liu, C. Chen, and B. Zeng. A hierarchical approach for rain or snow removing in a single color image. *IEEE Transactions on Image Processing*, 26(8):3936–3950, 2017.

[33] D.-W. Ren, W.-M. Zuo, D. Zhang, L. Zhang, and M.-H. Yang. Simultaneous fidelity and regularization learning for image restoration. *arXiv preprint arXiv:1804.04522*, 2018.

[34] X.-Y. Fu, J.-B. Huang, X.-H. Ding, Y.-H. Liao, and J. Paisley. Clearing the skies: A deep network architecture for single-image rain removal. *IEEE Transactions on Image Processing*, 26(6):2944–2956, 2017.

[35] H. Zhang, V. Sindagi, and V.-M Patel. Image de-raining using a conditional generative adversarial network. *arXiv preprint arXiv:1701.05957*, 2017.

[36] K. Garg and S.-K Nayar. Detection and removal of rain from videos. In *the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 1, 2004.

[37] A. Tripathi and S Mukhopadhyay. Video post processing: low-latency spatiotemporal approach for detection and removal of rain. *IET image processing*, 6(2):181–196, 2012.

[38] A.-K. Tripathi and S. Mukhopadhyay. Removal of rain from videos: a review. *Signal, Image and Video Processing*, 8(8):1421–1430, 2014.

[39] J.-H. Kim, J.-Y. Sim, and C.-S. Kim. Video deraining and desnowing using temporal correlation and low-rank matrix completion. *IEEE Transactions on Image Processing*, 24(9):2658–2670, 2015.

[40] V. Santhaseelan and V.-K Asari. Utilizing local phase information to remove rain from video. *International Journal of Computer Vision*, 112(1):71–89, 2015.

[41] S.-D. You, R.-T Tan, R. Kawakami, Y. Mukaigawa, and K. Ikeuchi. Adherent raindrop modeling, detectionand removal in video. *IEEE transactions on pattern analysis and machine intelligence*, 38(9):1721–1733, 2016.

[42] W.-H. Ren, J.-D. Tian, Z. Han, A. Chan, and Y.-D. Tang. Video desnowing and deraining based on matrix decomposition. In *the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4210–4219, 2017.

[43] L. Shen, Z.-H. Yue, Q. Chen, F. Feng, and J. Ma. Deep joint rain and haze removal from single images. *arXiv preprint arXiv:1801.06769*, 2018.

[44] J.-H. Kim, C. Lee, J.-Y. Sim, and C.-S. Kim. Single-image deraining using an adaptive nonlocal means filter. In *the IEEE International Conference on Image Processing (ICIP)*, pages 914–917, 2013.

[45] C.-H. Son and X.-P. Zhang. Rain removal via shrinkage-based sparse coding and learned rain dictionary. *arXiv preprint arXiv:1610.00386*, 2016.

[46] J. Chen and L.-P. Chau. A rain pixel recovery algorithm for videos with highly dynamic scenes. *IEEE Transactions on Image Processing*, 23(3):1097–1104, 2014.

[47] D. Eigen, D. Krishnan, and R. Fergus. Restoring an image taken through a window covered with dirt or rain. In *the IEEE International Conference onComputer Vision (ICCV)*, pages 633–640, 2013.

[48] K.-M. He, X.-Y. Zhang, S.-Q. Ren, and J. Sun. Deep residual learning for image recognition. In *the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[49] J.-S. Pan, S.-F. Liu, D.-Q. Sun, J.-W. Zhang, Y. Liu, J. Ren, Z.-C. Li, J.-H. Tang, H.-C. Lu, Y.-W. Tai, and et al. Learning dual convolutional neural networks for low-level vision. In *the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[50] Z.-W. Fan, H.-F. Wu, X.-Y. Fu, Y. Hunag, and X.-H. Ding. Residual-guide feature fusion network for single image deraining. *arXiv preprint arXiv:1804.07493*, 2018.

[51] X.-Y. Fu, B. Liang, Y. Huang, X.-H. Ding, and J. Paisley. Lightweight pyramid networks for image deraining. *arXiv preprint arXiv:1805.06173*, 2018.

[52] K.-M. He, J. Sun, and X.-O. Tang. Guided image filtering. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35:1397–1409, 2012.

[53] A. Krizhevsky, I. Sutskever, and G.-E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems (NIPS)*, pages 1097–1105, 2012.

[54] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International Conference on Machine Learning (ICML)*, pages 448–456, 2015.

[55] D.-P Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[56] K. Garg and S.-K Nayar. Photorealistic rendering of rain streaks. In *ACM Transactions on Graphics (TOG)*, volume 25, pages 996–1002. ACM, 2006.

[57] Z. Wang, A.-C Bovik, H.-R Sheikh, and E.-P Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13(4):600–612, 2004.

[58] L. Zhang, L. Zhang, X.-Q. Mou, and D. Zhang. Fsim: A feature similarity index for image quality assessment. *IEEE transactions on Image Processing*, 20(8):2378–2386, 2011.

[59] Z. Wang and A.-C Bovik. A universal image quality index. *IEEE Signal Processing Letters*, 9(3):81–84, 2002.

[60] W.-F. Xue, L. Zhang, X.-Q. Mou, and A.-C Bovik. Gradient magnitude similarity deviation: A highly efficient perceptual image quality index. *IEEE Transactions on Image Processing*, 23(2):684–695, 2014.

[61] Y. Li and M.-S. Brown. Single image layer separation using relative smoothness. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 2752–2759, 2014.

[62] L.-J. Deng, T.-Z. Huang, X.-L. Zhao, and T.-X. Jiang. A directional global sparse model for single image rain removal. *Applied Mathematical Modelling*, 59:662–679, 2018.