FastDeRain: A Novel Video Rain Streak Removal Method Using Directional Gradient Priors

Tai-Xiang Jiang, Ting-Zhu Huang, Xi-Le Zhao, Liang-Jian Deng, and Yao Wang

***Abstract*— Rain streaks removal is an important issue in outdoor vision systems and has recently been investigated extensively. In this paper, we propose a novel video rain streak removal approach FastDeRain, which fully considers the discriminative characteristics of rain streaks and the clean video in the gradient domain. Specifically, on the one hand, rain streaks are sparse and smooth along the direction of the raindrops, whereas on the other hand, clean videos exhibit piecewise smoothness along the rain-perpendicular direction and continuity along the temporal direction. Theses smoothness and continuity result in the sparse distribution in the different directional gradient domain. Thus, we minimize: 1) the ℓ_1 norm to enhance the sparsity of the underlying rain streaks; 2) two ℓ_1 norm of unidirectional total variation regularizers to guarantee the anisotropic spatial smoothness; and 3) an ℓ_1 norm of the time-directional difference operator to characterize the temporal continuity. A split augmented Lagrangian shrinkage algorithm-based algorithm is designed to solve the proposed minimization model. Experiments conducted on synthetic and real data demonstrate the effectiveness and efficiency of the proposed method. According to the comprehensive quantitative performance measures, our approach outperforms other state-of-the-art methods, especially on account of the running time. The code of FastDeRain can be downloaded at https://github.com/TaiXiangJiang/FastDeRain.**

***Index Terms*— Video rain streak removal, unidirectional total variation, split augmented Lagrangian shrinkage algorithm (SALSA).**

I. Introduction

OUTDOOR vision systems are frequently affected by bad weather conditions [1]–[5], one of which is the rain. Raindrops usually introduce bright streaks into the acquired images or videos, because of their scattering of light into complementary metal–oxide–semiconductor cameras and their high velocities. Moreover, rain streaks also interfere with nearby pixels because of their specular highlights, scattering, and blurring effects [1]. This undesirable interference will degrade the performance of various computer vision algorithms [6], such as event detection [7], object detection [8], tracking [9], recognition [10], and scene analysis [11]. Therefore, the removal of rain streaks is an essential task [78], which has recently received considerable attention.

Fig. 1. A frame of a rainy video (left), the rain streaks removal result by the proposed method FastDeRain (middle) and the extracted rain streaks (right). The pixel values of the rain streaks are scaled for better visualization.

Numerous methods have been proposed to improve the visibility of images/videos captured with rain streak interference [12]–[49]. They can be classified into two categories: multiple-images/videos based techniques and single-image based approaches. Fig. 1 exhibits an example of video rain streaks removal. Without loss of generality, in this paper, we use "background" to denote the rain-free content of the data.

For the single-image de-raining task, Kang *et al*. [12] 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*. [13] tactfully exploited the structural similarity of the derived HF image bases. Chen and Hsu [14] considered the similar and repeated patterns of the rain streaks and the smoothness of the background. Sparse coding and dictionary learning were adopted in [16]–[18]. In their results, the details of backgrounds were well preserved. Meanwhile, Zhang and Patel [19] decomposed a rainy image into a clear background image and a rain streak image using a set of generic sparsity-based and low-rank representation-based convolutional filters. The recent work by Li *et al*. [1], [20] utilized Gaussian mixture model (GMM)

Manuscript received March 20, 2018; revised July 15, 2018 and October 24, 2018; accepted October 26, 2018. Date of publication November 12, 2018; date of current version December 19, 2018. This work was supported in part by the National Natural Science Foundation of China under Grant 61772003, Grant 61876203, Grant 61702083, and Grant 11501440, in part by the Fundamental Research Funds for the Central Universities under Grant ZYGX2016J132, Grant ZYGX2016J129, and Grant ZYGX2016KYQD142, in part by the Science Strength Promotion Programme of UESTC, and in part by the Key Research Program of Hunan Province, China, under Grant 2017GK2273. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Xiao-Ping Zhang. *(Corresponding authors: Ting-Zhu Huang; Xi-Le Zhao.)*

T.-X Jiang, T.-Z. Huang, X.-L. Zhao, and L.-J. Deng are with the School of Mathematical Sciences, University of Electronic Science and Technology of China, Chengdu 611731, China (e-mail: taixiangjiang@gmail.com; tingzhuhuang@126.com; xlzhao122003@163.com; liangjian1987112@126.com).

Y. Wang is with the School of Mathematics and Statistics, Xi'an Jiaotong University, Xian 710049, China (e-mail: yao.s.wang@gmail.com).

Digital Object Identifier 10.1109/TIP.2018.2880512

patch priors for rain streak removal, with the ability to account for rain streaks of different orientations and scales. Zhu *et al*. [21] proposed a joint bi-layer optimization method progressively separate rain streaks from background details, in which the gradient statistics are analyzed. Meanwhile, the directional property of rain streaks received a lot of attention in [24]–[26] and these methods achieved promising performances. Wang *et al*. [27] took advantage the image decomposition and dictionary learning. The recently developed deep learning technique was also applied to the single image rain streaks removal task, and excellent results were obtained [28]–[38].

For the video rain streaks removal, Garg and Nayar [39] firstly raised a video rain streaks removal method with comprehensive analysis of the visual effects of the rain on an imaging system. Since then, many approaches have been proposed for the video rain streaks task and obtained good rain removing performance in videos with different rain circumstances. Comprehensive early existing video-based methods are summarized in [40]. Chen and Chau [15] took account of the highly dynamic scenes. Whereafter, Kim *et al*. [41] considered the temporal correlation of rain streaks and the low-rank nature of clean videos. Santhaseelan and Asari [42] detected and removed the rain streaks based on phase congruency features. You *et al*. [43] dealt with the situations where the raindrops are adhered to the windscreen or the window glass. In [44], a novel tensor-based video rain streak removal approach was proposed considering the directional property. Ren *et al*. [45] handled the video desnowing and deraining task based on matrix decomposition. The rain streaks and the clean background were stochastically modeled as a mixture of Gaussians by Wei *et al*. [46] while Li *et al*. [47] learned the multiscale convolutional filters from the rainy data. Both of these two methods [46], [47] achieved excellent performances with surveillance videos. For the video rain streaks removal, the deep learning based methods also started to reveal their effectiveness [48]–[50].

In general, the observation model for a rainy image is formulated as $\mathbf{O} = \mathbf{B} + \mathbf{R}$ [1], which can be generalized to the video case as: $\mathcal{O} = \mathcal{B} + \mathcal{R}$, where $\mathcal{O}$, $\mathcal{B}$, and $\mathcal{R} \in \mathbb{R}^{m\times n\times t}$ are three 3-mode tensors representing the observed rainy video, the unknown rain-free video and the rain streaks, respectively. When considering the noise or error, the observation model is modified as $\mathcal{O} = \mathcal{B} + \mathcal{R} + \mathcal{N}$, where $\mathcal{N}$ is the noise or error term. The goal of video rain streak removal is to distinguish the clean video $\mathcal{B}$ and the rain streaks $\mathcal{R}$ from an input rainy video $\mathcal{O}$. This is an ill-posed inverse problem, which can be handled by imposing prior information. Therefore, from this point of view, the most significant issues are the rational extraction and sufficient utilization of the prior knowledge, which is helpful to wipe off the rain streaks and reconstruct the rain-free video. In this paper, we mainly focus on the discriminative characteristics of rain streaks and background in different directional gradient domains.

From the temporal perspective, the clean video is continuous along the time direction, while the rain streaks do not share this property [41], [46], [51]. As observed in Fig. 2, the time-directional gradient of the rain-free video (a-2) exhibits a different histogram compared with those of the rainy video (a-1) and the rain streaks (a-3). The temporal gradient of the clean video is much sparser and it is corresponding to the temporal continuity of the clean video. Therefore, we intend to minimize $\|\nabla_t \mathcal{B}\|_1$, where ∇_t is the temporal differential operator.

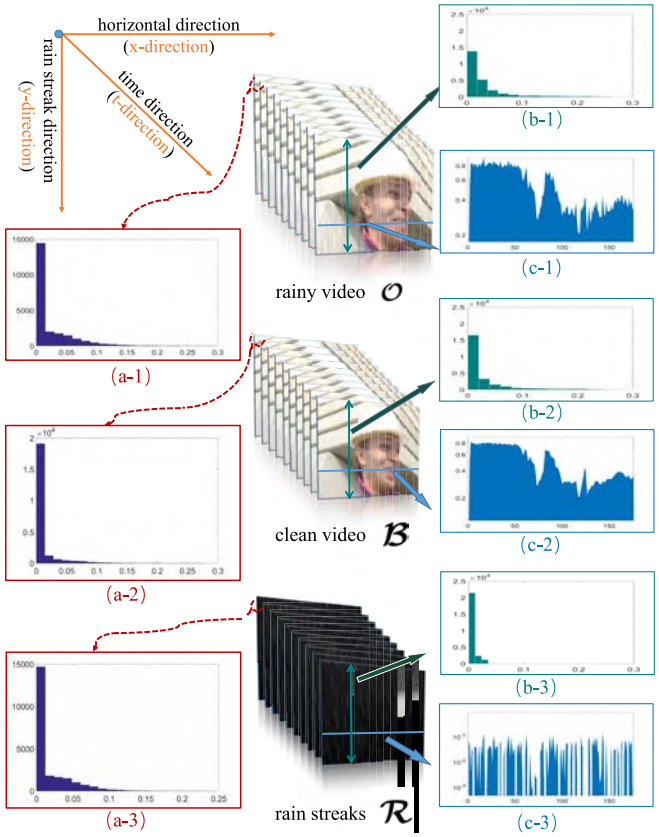


Fig. 2. From left to right: the histograms of temporal gradient of the rainy video (a-1), the clean video (a-2) and the isolated rain streaks (a-3), respectively; several example frames from the rainy video, the clean video and the isolated rain streaks; and the histograms of the vertical gradient (b-1,2,3) and the intensities along a row (c-1,2,3) in the rainy video, the clean video and the isolated rain streaks, respectively.

From the spatial perspective, it has been widely recognized that natural images are largely piecewise smooth and their gradient fields or the coefficients in the tight wavelet frame domain are typically sparse [52]–[54]. Many aforementioned de-rain methods take the spatial gradient into consideration and use the total variation (TV) to depict the property of the rain-free part [1], [14]. However, the effects of the rain streaks on the vertical gradient and horizontal gradient are different. This phenomenon was likewise noticed in [24]–[26]. Initially, for the sake of convenience, we assume that rain streaks are approximately vertical. The impact of the vertical rain streaks on the vertical gradient is limited. The subfigures (b-1,2,3) in Fig. 2 reveal that the vertical gradient of rain streaks are much sparser than those of the clean video and the rainy video. Nonetheless, the vertical rain streaks severely disrupt the horizontal piecewise smoothness. As exhibited in Fig. 2 (c-1,2,3), the pixel intensity is piecewise smooth

only in (c-2), whereas burrs frequently appear in (c-1) and (c-3). Therefore, we intend to minimize $\|\nabla_1\mathcal{R}\|_1$ and $\|\nabla_2\mathcal{B}\|_1$, where ∇_1 and ∇_2 are respectively the vertical difference (or say vertical unidirectional TV [55]–[57]) operator and horizontal difference (or say horizontal unidirectional TV) operator.

Given a real rainfall-affected scene, without the wind, the raindrops generally fall from top to bottom. Meanwhile, when not very windy, the angles between rain streaks and the vertical direction are usually not very large. Therefore, the rain streak direction can be approximated as the vertical direction, *i.e.* the mode-1 (column) direction of the video tensor. Actually, this assumption is reasonable for parts of the rainy sceneries. For the rain streaks that are oblique (or say far from being vertical), directly utilizing the directional property is very difficult for the digital video data, which are cubes of distinct numbers. To cope with this difficulty, in Sec. III-E, we would design the shift strategy, based on our automatical rain streaks' direction detection method.

The contributions of this paper include three aspects.

- We propose a video rain streaks removal model, which fully considers the discriminative prior knowledge of the rain streaks and the clean video.
- We design a split augmented Lagrangian shrinkage algorithm (SALSA) based algorithm to efficiently and effectively solve the proposed minimization model. The convergence of our algorithm is theoretically guaranteed. Meanwhile, the implementation on the graphics processing unit (GPU) device further accelerates our method.
- To demonstrate the efficacy and the superior performance of the proposed algorithm in comparison with state-of-the-art alternatives, extensive experiments both on the synthetic data and the real-world rainy videos are conducted.

This work is an extension of the material published in [44]. The new material is the following: a) the proposed rain streaks removal model is improved and herein introduced in more technical details; b) we explicitly use the split augmented Lagrangian shrinkage algorithm to solve the proposed model; c) to make the proposed method more applicable, we provide the shift strategy to deal with oblique rain streaks; d) in our experiments, we re-simulate the rain streaks for the synthetic data, using two different techniques and considering the rain streaks not very vertical; e) three recent state-of-the-art methods, consisting of methods in [31] and [47] and the method in our conference paper [44], are brought into comparison.

The paper organized as follows. Section II gives the preliminary on the tensor notations. In Section III, the formulation of our model is presented along with a SALSA solver. Experimental results are reported in Section IV. Finally, we draw some conclusions in Section V.

II. Notation and Preliminaries

Following [58]–[61], we use lower-case letters for vectors, e.g., **a**; upper-case letters for matrices, e.g., **A**; and calligraphic letters for tensors, e.g., $\mathcal{A}$. An N-mode tensor is defined as $\mathcal{X} \in \mathbb{R}^{I_1\times I_2\times\cdots\times I_N}$, and $x_{i_1,i_2,\cdots,i_N}$ denotes its $(i_1, i_2, \cdots, i_N)$-th component.

TABLE I
Tensor Notations

Notation	Explanation
$\mathcal{X}, \mathbf{X}, \mathbf{x}, x$	Tensor, matrix, vector, scalar.
$\mathbf{x}(:i_2i_3\cdots i_N)$	A **fiber** of a tensor $\mathcal{X}$, defined by fixing every index but one.
$\mathbf{X}(::i_3\cdots i_N)$	A **slice** of a tensor $\mathcal{X}$, defined by fixing all but two indices.
$\langle\mathcal{X},\mathcal{Y}\rangle$	The **inner product** of two same-sized tensors $\mathcal{X}$ and $\mathcal{Y}$.
$\|\mathcal{X}\|_F$	The **Frobenius norm** of a tensor $\mathcal{X}$.

A **fiber** of a tensor is defined by fixing every index but one. A third-order tensor has column, row, and tube fibers, denoted by $\mathbf{x}_{:jk}$, $\mathbf{x}_{i:k}$, and $\mathbf{x}_{ij:}$, respectively. When extracted from their tensors, fibers are always assumed to be oriented as column vectors.

A **slice** is a two-dimensional section of a tensor, defined by fixing all but two indices. The horizontal, lateral, and frontal slides of a third-order tensor $\mathcal{X}$ are denoted by $\mathbf{X}_{i::}$, $\mathbf{X}_{:j:}$, and $\mathbf{X}_{::k}$, respectively. Alternatively, the k-th frontal slice of a third-order tensor, $\mathbf{X}_{::k}$, may be denoted more compactly by $\mathbf{X}_k$.

The **inner product** of two same-sized tensors $\mathcal{X}$ and $\mathcal{Y}$ is defined as $\langle\mathcal{X},\mathcal{Y}\rangle := \sum_{i_1,i_2,\cdots,i_N} x_{i_1i_2\cdots i_N}\cdot y_{i_1i_2\cdots i_N}$. The corresponding norm (**Frobenius norm**) is then defined as $\|\mathcal{X}\|_F := \sqrt{\langle\mathcal{X},\mathcal{X}\rangle}$.

Please refer to [62] for a more extensive overview.

III. Main Results

A. Problem Formulation

As mentioned before, a rainy video $\mathcal{O} \in \mathbb{R}^{m\times n\times t}$ can be modeled as a linear superposition:

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

where $\mathcal{O}, \mathcal{B}, \mathcal{R}$ and $\mathcal{N} \in \mathbb{R}^{m\times n\times t}$ are four 3-mode tensors representing the observed rainy video, the unknown rain-free video, the rain streaks and the noise (or error) term, respectively.

Our goal is to decompose the rain-free video $\mathcal{B}$ and the rain streaks $\mathcal{R}$ from an input rainy video $\mathcal{O}$. To solve this ill-posed inverse problem, we need to analyze the prior information for both $\mathcal{B}$ and $\mathcal{R}$ and then introduce corresponding regularizers, which will be discussed in the next subsection.

B. Priors and Regularizers

In this subsection, we continue the discussion on the prior knowledge with the assumption that rain streaks are approximately vertical.

a) Sparsity of rain streaks: When the rain is light, the rain streaks can naturally be considered as being sparse. To boost the sparsity of rain streaks, minimizing the ℓ_1 norm of the rain streaks $\mathcal{R}$ is an ideal option. When the rain is very heavy, it seems that this regularization is not proper. However, when

the rain is extremely heavy, it is very difficult or even impossible to recover the rain-free part because of the huge loss of the reliable information. The rainy scenarios discussed in this paper are not that extreme, and we assume that the rain streaks always maintain lower energy than the background clean videos. Therefore, when the rain streaks are dense, the ℓ_1 norm can be viewed as a role to restrain the magnitude of the rain streaks. Meanwhile, in our model, other regularization terms would also contribute to distinguishing the rain streaks. Thus, we can tackle the heavy raining scenarios by tuning the parameter of the sparsity term so as to reduce its effect.

b) The horizontal direction: In Fig. 2, (c-1,2,3) show the pixel intensities along a fixed row of the rainy video, the clean video and the rain streaks, respectively. It is obvious that the variation of the pixel intensity is piecewise smooth only in (c-2), whereas burrs frequently appear in (c-1) and (c-3). Therefore, a horizontal unidirectional TV regularizer is a suitable candidate for $\mathcal{B}$.

c) The vertical direction: It can be seen from Fig. 2 that (b-3), which is the histogram of the intensity of the vertical gradient in a rain-streak frame, exhibits a distinct distribution with respect to (c-1) and (c-2). The long-tailed distributions in (c-1) and (c-3) indicate that the minimization of the l_1 norm of $\boldsymbol{\nabla}_1\mathcal{R}$ would help to distinguish the rain streaks.

d) The temporal direction: From the first column of Fig. 2, it can be observed that clean videos exhibit the continuity along the time axis. Sub-figures (a-1,2,3), which present the histograms of the magnitudes in the temporal directional gradient, illustrate that the clean video's temporal gradients consist of more zero values and smaller non-zero values, whereas those of the rainy video and rain streaks tend to be long-tailed. Therefore, it is natural to minimize the l_1 norm of the temporal gradient of the clean video $\mathcal{B}$. By the way, the low-rank regularization used in [44] is discarded since that the low-rank assumption is not reasonable for the videos captured by dynamic cameras and the rain streaks, which always share the repetitive patterns, can occasionally be more low-rank than the background along the spatial directions.

C. The Proposed Model

Generally, there is an angle between the vertical direction and the real falling direction of the raindrops. The rain streaks pictured in Fig. 2 are not strictly vertical and there is a 5-degree angle between the rain streaks and the y-axis. In other words, the prior knowledge discussed above are still valid when this angle is small. Large-angle cases would be discussed in Sec. III-E). Therefore, the rain streak direction is referred to as the vertical direction corresponding to the y-axis, whereas the rain-perpendicular direction is referred to as the horizontal direction corresponding to the x-axis. Thus, as a summary of the discussion of the priors and regularizers, our model can be compactly formulated as follows:

$$\begin{aligned}\min_{\mathcal{B},\mathcal{R}}\ & \alpha_1\|\boldsymbol{\nabla}_1\mathcal{R}\|_1+\alpha_2\|\mathcal{R}\|_1+\alpha_3\|\boldsymbol{\nabla}_2\mathcal{B}\|_1\\ &+\alpha_4\|\boldsymbol{\nabla}_t\mathcal{B}\|_1+\frac{1}{2}\|\mathcal{O}-(\mathcal{B}+\mathcal{R})\|_F^2\\ \text{s.t. }& \mathcal{O}\geqslant\mathcal{B}\geqslant\mathbf{0},\ \mathcal{O}\geqslant\mathcal{R}\geqslant\mathbf{0},\end{aligned}\tag{2}$$

where ∇_1, ∇_2 and ∇_t are the vertical, horizontal and temporal differential operators, respectively. ∇_1 and ∇_2 are also written as ∇_y and ∇_x in [24] and [44]. An efficient algorithm is proposed in the following subsection to solve (2).

D. Optimization

Since the proposed model (2) is concise and convex, many state-of-the-art solvers are available to solve it. Here, we apply the ADMM [63], which has been proved an effective strategy for solving large scale optimization problems [64]–[67]. More specifically, we adopt SALSA [68].

After introducing four auxiliary tensors the proposed model (2) is reformulated as the following equivalent constrained problem:

$$\begin{aligned}\min_{\mathcal{B},\mathcal{V}_i,\mathcal{D}_i}\ & \alpha_1\|\mathcal{V}_1\|_1+\alpha_2\|\mathcal{V}_2\|_1+\alpha_3\|\mathcal{V}_3\|_1+\alpha_4\|\mathcal{V}_4\|_1\\ &+\frac{1}{2}\|\mathcal{O}-(\mathcal{B}+\mathcal{R})\|_F^2\\ \text{s.t. }& \mathcal{V}_1=\boldsymbol{\nabla}_1(\mathcal{R}),\quad \mathcal{V}_2=\mathcal{R},\ \mathcal{V}_3=\boldsymbol{\nabla}_2(\mathcal{B}),\\ & \mathcal{V}_4=\boldsymbol{\nabla}_t(\mathcal{B}),\quad \mathcal{O}\geqslant\mathcal{B}\geqslant\mathbf{0},\ \mathcal{O}\geqslant\mathcal{R}\geqslant\mathbf{0}\end{aligned}\tag{3}$$

where $\mathcal{V}_i\ \in\mathbb{R}^{m\times n\times t}$ $(i=1,2,3,4)$.

Then, the augmented Lagrangian function of (3) is

$$\begin{aligned}&L_\mu(\mathcal{B},\mathcal{R},\mathcal{V}_i,\mathcal{D}_i)\\ &=\frac{1}{2}\|\mathcal{O}-\mathcal{B}-\mathcal{R}\|_F^2+\alpha_1\|\mathcal{V}_1\|_1+\alpha_2\|\mathcal{V}_2\|_1\\ &\quad+\alpha_3\|\mathcal{V}_3\|_1+\alpha_4\|\mathcal{V}_4\|_1+\frac{\mu}{2}\|\boldsymbol{\nabla}_1\mathcal{R}-\mathcal{V}_1-\mathcal{D}_1\|_F^2\\ &\quad+\frac{\mu}{2}\|\mathcal{R}-\mathcal{V}_2-\mathcal{D}_2\|_F^2+\frac{\mu}{2}\|\boldsymbol{\nabla}_2\mathcal{B}-\mathcal{V}_3-\mathcal{D}_3\|_F^2\\ &\quad+\frac{\mu}{2}\|\boldsymbol{\nabla}_t\mathcal{B}-\mathcal{V}_4-\mathcal{D}_4\|_F^2,\end{aligned}$$

where the $\mathcal{D}_i$s $(i=1,2,3,4)$ are the scaled Lagrange multipliers and the μ is a positive scalar.

e) $\mathcal{V}_i$ sub-problems: For $i=1,2,3,4$, the $\mathcal{V}_i$ sub-problem can be written as a equivalent problem:

$$\mathcal{V}_i^+=\arg\min_{\mathcal{V}_i}\ \alpha_i\|\mathcal{V}_i\|_1+\frac{\mu}{2}\|\mathcal{A}_i-\mathcal{V}_i\|_F^2.$$

Such a problem has a closed-form solution, obtained through soft thresholding:

$$\mathcal{V}_i^+=\boldsymbol{\mathcal{S}}_{\frac{\alpha_i}{\mu}}(\mathcal{A}_i).$$

Here, the tensor non-negative **soft-thresholding operator** $\boldsymbol{\mathcal{S}}_\upsilon(\cdot)$ is defined as

$$\boldsymbol{\mathcal{S}}_\upsilon(\mathcal{A})=\bar{\mathcal{A}}$$

with

$$\bar{a}_{i_1i_2\cdots i_N}=\begin{cases}a_{i_1i_2\cdots i_N}-\upsilon, & a_{i_1i_2\cdots i_N}>\upsilon,\\ 0, & \text{otherwise.}\end{cases}$$

Therefore, $\mathcal{V}_i$ $(i = 1, 2, 3, 4)$ can respectively be updated as follows:

$$\begin{cases} \mathcal{V}_1^{(t+1)} = \boldsymbol{\mathcal{S}}_{\frac{\alpha_1}{\mu}}(\nabla_1\mathcal{R} - \mathcal{D}_1), \\ \mathcal{V}_2^{(t+1)} = \boldsymbol{\mathcal{S}}_{\frac{\alpha_2}{\mu}}(\mathcal{R} - \mathcal{D}_2), \\ \mathcal{V}_3^{(t+1)} = \boldsymbol{\mathcal{S}}_{\frac{\alpha_3}{\mu}}(\nabla_2\mathcal{B} - \mathcal{D}_3), \\ \mathcal{V}_4^{(t+1)} = \boldsymbol{\mathcal{S}}_{\frac{\alpha_4}{\mu}}(\nabla_t\mathcal{B} - \mathcal{D}_4). \end{cases} \tag{4}$$

The time complexity of each sub-problem above is $O(mnt)$.

f) $\mathcal{B}$ and $\mathcal{R}$ sub-problems: $\mathcal{B}$ and $\mathcal{R}$ sub-problems are least-squares problems:

$$\begin{aligned} \mathcal{B}^+ = \arg\min_{\mathcal{O}\le\mathcal{B}\le\mathbf{0}} & \frac{1}{2}\|\mathcal{O} - \mathcal{B} - \mathcal{R}\|_F^2 + \frac{\mu}{2}\|\nabla_2\mathcal{B} - \mathcal{V}_3 - \mathcal{D}_3\|_F^2 \\ & + \frac{\mu}{2}\|\nabla_t\mathcal{B} - \mathcal{V}_4 - \mathcal{D}_4\|_F^2, \\ \mathcal{R}^+ = \arg\min_{\mathcal{O}\le\mathcal{R}\le\mathbf{0}} & \frac{1}{2}\|\mathcal{O} - \mathcal{B} - \mathcal{R}\|_F^2 + \frac{\mu}{2}\|\nabla_1\mathcal{R} - \mathcal{V}_1 - \mathcal{D}_1\|_F^2 \\ & + \frac{\mu}{2}\|\mathcal{R} - \mathcal{V}_2 - \mathcal{D}_2\|_F^2. \end{aligned}$$

Then, we have

$$\begin{aligned} \mathcal{B}^+ &= \frac{\mathcal{O} - \mathcal{R} + \mu\nabla_2^\top(\mathcal{V}_3 - \mathcal{D}_3) + \mu\nabla_t^\top(\mathcal{V}_4 - \mathcal{D}_4)}{\mathbf{1} + \mu\nabla_2^\top\nabla_2 + \mu\nabla_t^\top\nabla_t} \\ \mathcal{R}^+ &= \frac{\mathcal{O} - \mathcal{B} + \mu\nabla_1^\top(\mathcal{V}_1 - \mathcal{D}_1) + \mu(\mathcal{V}_2 - \mathcal{D}_2)}{\mathbf{1} + \mu\nabla_1^\top\nabla_1 + \mu} \end{aligned} \tag{5}$$

We adopt the fast Fourier transform (FFT) for fast calculation when updating $\mathcal{B}$ and $\mathcal{R}$. Meanwhile, the elements in $\mathcal{B}^{(t+1)}$ and $\mathcal{R}^{(t+1)}$that are smaller than 0 or larger than the corresponding elements in $\mathcal{O}$ will be shrunk. The time complexity of updating $\mathcal{B}$ (or $\mathcal{R}$) is $O(mnt \cdot \log(mnt))$.

g) Multipliers updating: The Lagrange multipliers $\mathcal{D}_i$s $(i = 1, 2, 3, 4)$ can be updated as follows:

$$\begin{cases} \mathcal{D}_1 = \mathcal{D}_1 + \nabla_1\mathcal{R} - \mathcal{V}_1 \\ \mathcal{D}_2 = \mathcal{D}_2 + \mathcal{R} - \mathcal{V}_2 \\ \mathcal{D}_3 = \mathcal{D}_3 + \nabla_2\mathcal{B} - \mathcal{V}_3 \\ \mathcal{D}_4 = \mathcal{D}_4 + \nabla_t\mathcal{B} - \mathcal{V}_4 \end{cases} \tag{6}$$

The proposed algorithm for video rain streak removal is denoted as "FastDeRain" and summarized in Algorithm 1. For a video with dimensions of $m \times n \times t$, the time complexity of the proposed algorithm is proportional to $O(mnt\log(mnt))$.

Algorithm 1 FastDeRain

Input: The rainy video $\mathcal{O}$;
Initialization: $\mathcal{B}^{(0)} = \mathcal{O}$, $\mathcal{O} = \mathbf{0}$
1: **while** not converged **do**
2: Update $\mathcal{V}_i$ $(i = 1, 2, 3, 4)$ via Eq. (4);
3: Update $\mathcal{B}$ and $\mathcal{R}$ via (5);
4: Update $\mathcal{D}_i$ $(i = 1, 2, 3, 4)$ via Eq. (6);
5: **end while**
Output: The estimates of the rain-free video $\mathcal{B}$ and the rain streaks $\mathcal{R}$.

Fig. 3. Illustrations of the shift I and the shift II operations. For better visualization, the rain streaks in the left part are roughly labeled with the red color, while the pixel values of the rain streaks images in the right are scaled.

E. The Shift Strategy for Oblique Rain Streaks

As we know that, in a real rainfall-affected scene, the rain streaks are not always vertical. Thus, the directional property we utilized in our model is a double-edged sword when dealing with digital videos. Fortunately, as shown in the experimental part, our FastDeRain is robust to a range of angles, about $-15°$ to $15°$ with respect to the vertical direction. Thus, we consider to divide the rainy situations into different cases.

Without loss of generality, we assume that the rain streaks are in a similar direction and the angle between rain streaks and the vertical direction is denoted as θ. Generally, the angle θ distributes in $(-90°, 90°)$. If the angle $\theta \in (-90°, 0°)$, we can restrict it to the range of $(0°, 90°)$ by the left-right flipping of each frame. When $\theta \in (45°, 90°)$, we can restrict it to the range of $(0°, 45°)$ by transposing (i.e. interchanging the rows and columns of a given matrix) each frame. Therefore, our goal turns to handle the rain streaks with angles in $[0°, 45°]$.

When the angle θ is close to zero and not bigger than, we directly use our FastDeRain, and for other elaborately In this subsection, inspired by the shearing techniques used for cartoon-texture image decomposition [69], we propose the shift strategy to deal with rain streaks not vertical.

a) The shift operations: We first introduce two shift operations, as shown in Fig. 3. Different from the rotation operation recommended in [44], the core idea of the shift operations is to rationally slide the rows of the rainy frames and make the rain streaks being approximately vertical without any degradation caused by the interpolation [70]. We remark here that these shifting operations would not affect the priors mentioned in Sec. III-B. These two shift operations are detailed as follows:

Shift I For each frame $\mathbf{O}_{::k}$, we slide the i-th row $(i-1)$ pixel(s) to the right.

Shift II For each frame $\mathbf{O}_{::k}$, we slide the i-th row $\lfloor\frac{(i-1)}{2}\rfloor$[1] pixel(s) to the right.

Without loss of generality, we assume that the rain streaks are in a similar direction and the angle between rain streaks and the vertical direction is denoted as θ. Shift I is suitable for the rain streaks with $\theta = 45°$ while Shift II is ideal for $\theta = 26.57°$, since that $\arctan 1 = 45°$ and $\arctan 1/2 \approx 26.57°$. Considering that the proposed FastDeRain is robust to a range

[1] $\lfloor x \rfloor$ denotes the rounding the x to the nearest integers towards minus infinity.

of angles (see details in Sec. IV), our method with Shift I and Shift II is sufficient for the situations when $\theta \in [0°, 45°]$.

Generally, the angle θ distributes in $(-90°, 90°)$. If the angle $\theta \in (-90°, 0°)$, we can restrict it to the range of $(0°, 90°)$ by the left-right flipping of each frame. When $\theta \in (45°, 90°)$, we can restrict it to the range of $(0°, 45°)$ by transposing (i.e. interchanging the rows and columns of a given matrix) each frame. Hence, our method with the shift operations is able to handle all the cases.

b) The shift strategy: After giving the shift operations and the left-right flipping and transposing transformations, the question comes to how to automatically decide the transformation and the shift operation. Fortunately, based on our analysis of the prior knowledge in Sec. III-B, it's not difficult to come up with a practical and efficient strategy with these two shift operations.

Then, for a rainy video $\mathcal{O} \in \mathbb{R}^{m\times n\times t}$, our strategy consists of three steps:

1) Filtering. Filter the horizontal slices of the rainy video with a 3×3 median filter, *i.e.*, for $i = 1, 2, \cdots, m$, $\widehat{\mathcal{O}}(i, :, :) = \text{med}(\mathcal{O}(i, :, :))$, and obtain $\mathcal{R}_0 = \mathcal{O} - \widehat{\mathcal{O}}$.

2) Transforming and shifting. Left-right flip and transpose each frame of $\mathcal{R}_0$, and respectively apply shift I and shift II operations. Then we obtain a set of tensors, consisted of $\mathcal{R}_0$ and the variants of $\mathcal{R}_0$.

3) Computing vertical gradients. For each tensor $\mathcal{R}_0^i$ in this set, we compute $y_i = \|\nabla_1 \mathcal{R}_0^i\|_1$. Then we select the transformation and shift operations corresponding to the minimal y_i. By these three steps, the transforming and shift operations are automatically selected. We input the data after transforming and shifting into Algorithm 1 and finally conduct the inverse transformation and shift on the output.

IV. Experimental Results

In this section, we evaluate the performance of the proposed algorithm on synthetic data and real-world rainy videos.

a) Implementation details: Throughout our experiments, color videos with dimensions of $m \times n \times 3 \times t$ are transformed into the YUV format. YUV is a color space that is often used as part of a color image pipeline. Y stands for the luma component (the brightness), and U and V are the chrominance (color) components[2]. We apply our method only to the Y channel with the dimension of $m \times n \times t$. The exhibited rain streaks are scaled for better visualization.

Since that the graphics processing unit (GPU) device is able to speed up the large-scale computing, we implement our method on the platform of Windows 10 and Matlab (R2017a) with an Intel(R) Core(TM) i5-4590 CPU at 3.30GHz, 16 GB RAM, and a GTX1080 GPU. The involved operations in algorithm 1 is convenient to be implemented on the GPU device [71]. If we conduct our algorithm on the CPU, the running time for dealing with a video of size $240 \times 320 \times 3 \times 100$ is about 23 seconds, while 7 seconds on the GPU device. Meanwhile, Fu *et al.*'s method [31] can also be accelerated by the GPU device, from 38 seconds on the CPU to 24 seconds on the GPU, dealing with a video of size $240 \times 320 \times 3 \times 100$.

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

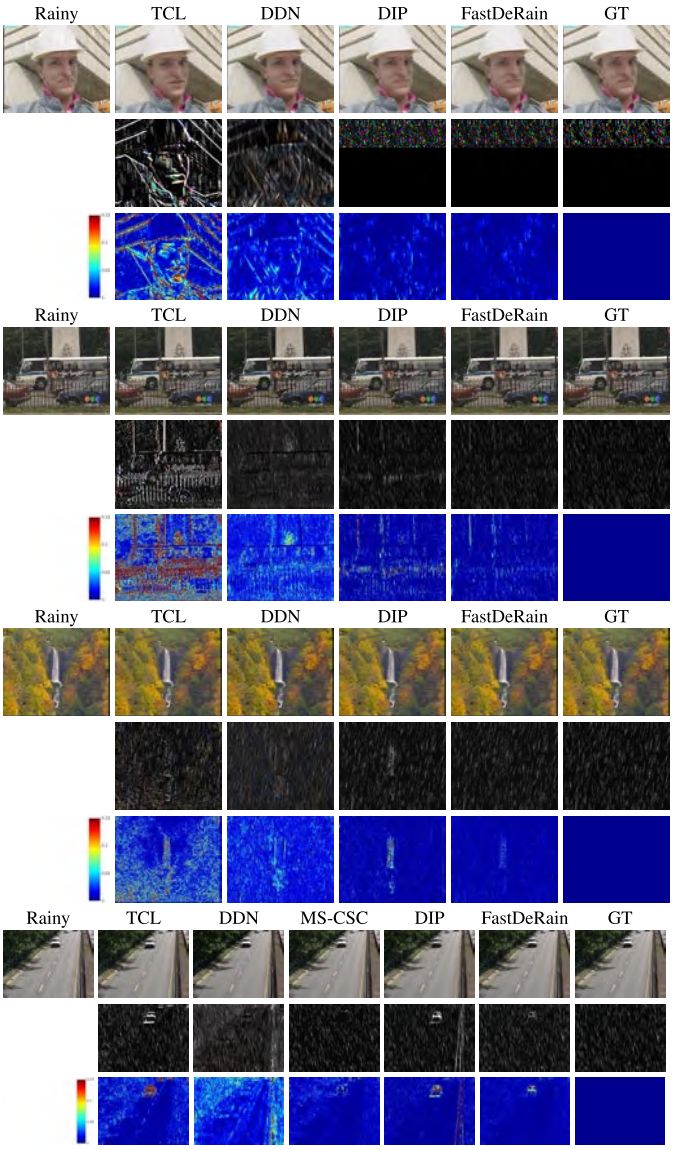


Fig. 4. The rainy frame, rain streaks removal results, extracted rain streaks and corresponding error images by different methods with synthetic rain streaks in **case 1**, respectively. The corresponding videos from top to bottom are the "'foreman", "bus", "waterfall" and "highway". From left to right are: the rainy data (or the color bar), results by TCL [41], DDN [31], DIP [44], (MS-CSC [47],) FastDeRain, and the ground truth (GT) clean video, respectively.

Thus, we only report the GPU running time of FastDeRain and Fu *et al.*'s method in this section.

b) Compared methods: To validate the effectiveness and efficiency of the proposed method, we compare our method (denoted as "FastDeRain") with recent state-of-the-art methods, including one single image based method, i.e., Fu *et al.*'s deep detail network (DDN) method[3] [31]; and three video-based methods, i.e., Kim *et al.*'s method using temporal correlation and low-rankness (TCL) [4] [41], the method, in our conference paper, utilizing discriminative intrinsic priors (DIP) [44], and Li *et al.*'s multiscale convolutional sparse cod-

[3]http://smartdsp.xmu.edu.cn/xyfu.html

[4]http://mcl.korea.ac.kr/jhkim/deraining/deraining_code_with_example.zip

ing (MS-CSC) method[5] [47]. In fact, DDN is a single-image-based rain streak removal method, but their performance has already surpassed some video-based methods. The deep learning technique shows a great vitality and an extremely wide application prospect. Hence, the comparison with DDN is reasonable and challenging.

A. Synthetic Data

a) Rain streak generation: Adding rain streaks to a video is indeed a complex problem since there is not an existing algorithm nor a free software to accomplish it in one step. Meanwhile, as Starik and Werman [51] pointed out that the rain streaks can be assumed temporal independent, thus we can simulate rain streaks for each frame using the synthetic method mentioned in many recently developed single image rain streaks removal approaches [12], [17], [30], *i.e.*, using the Photoshop software with the tutorial documents [72]. The density of the simulated rain streaks by this method is mainly determined by the ratio of the amounts of dots (in [72, Step 8]) to the number of all the pixels, and, for convenience, the ratio is denoted as r. Another way to synthesize the rain streaks was proposed in [46] and [47], adding rain streaks taken by photographers under black background[6].

Referring to [46], [47], and [72], we generate 3 types of rain streaks as follows:

Case 1: Rain streaks simulated referring to [72] with $r \leq 0.04$. In a single frame, the rain streaks share the same angle. The fixed angles for different frames increase from $-15°$ to $15°$ with time;

Case 2: Rain streaks simulated referring to [72] with $r \geq 0.05$. In a single frame, the rain streaks are with different angles. The angles uniformly distribute in a range $[-15°, 15°]$;

Case 3: Rain streaks simulated referring to [46].

Four videos are selected as the clean background. Three videos[7], named "foreman" with the size of $144 \times 176 \times 3 \times 160$, "bus" and "waterfall" with the size of $288 \times 352 \times 3 \times 100$, are captured by dynamic cameras, while the other one[8], named " highway" with the size of $240 \times 320 \times 3 \times 100$, are recorded by a static camera.

MS-CSC [47] is designed mainly for the videos captured by static cameras, and directly applying it on the video captured by dynamic camera would result in poor performances (see the gray values in Table II. Therefore, for a fair comparison, the compared methods included DDN [30], TCL [41] and DIP [44] when dealing with the synthetic rainy data generated on the videos "foreman" "bus" and "waterfall". When dealing with the rainy data simulated with the video "highway", MS-CSC [47] would be brought into comparison.

b) Quantitative comparisons: For quantitative assessment, the peak signal-to-noise ratio (PSNR) of the whole video, and the structural similarity (SSIM) [73], the feature similarity (FSIM) [74], the visual information fidelity (VIF) [75], the universal image quality index (UIQI) [76], and the gradient magnitude similarity deviation (GMSD, smaller is better) [77] of each frame are calculated. The PSNR, the corresponding mean values of SSIM FSIM VIF and UIQI, and the running time are reported in Table II, in which the best quantitative values are in boldface.

TABLE II

QUANTITATIVE COMPARISONS OF THE RAIN STREAK REMOVAL RESULTS OF [41], [31], [46], [47] AND THE PROPOSED METHOD ON SYNTHETIC VIDEOS. THE **BEST** QUANTITATIVE VALUES ARE IN **BOLDFACE**

	Video	Method	PSNR	SSIM	FSIM	VIF	UIQI	GMSD	Time
Case 1	*foreman*	Rainy	33.84	0.9511	0.9698	0.7457	0.8590	0.0656	0.0
		TCL [41]	33.41	0.9595	0.9701	0.6783	0.8919	0.0429	1703.0
		DDN [31]	34.15	0.9696	0.9779	0.7555	0.9000	0.0369	71.2
		MS-CSC [47]	26.64	0.7927	0.8773	0.3877	0.5681	0.1470	343.9
		DIP [44]	36.49	0.9776	0.9821	0.7826	**0.9374**	0.0326	22.8
		FastDeRain	**38.75**	**0.9823**	**0.9875**	**0.8047**	0.9285	**0.0205**	**2.6**
	"bus"	Rainy	31.01	0.9146	0.9664	0.7029	0.8806	0.0725	0.0
		TCL [41]	33.07	0.9562	0.9744	0.6976	0.9351	0.0360	3012.0
		DDN [31]	31.08	0.9534	0.9714	0.7132	0.9268	0.0399	46.4
		DIP [44]	31.20	0.9594	0.9721	0.7494	0.9400	0.0450	44.4
		FastDeRain	**35.96**	**0.9729**	**0.9849**	**0.7923**	**0.9546**	**0.0292**	**5.7**
	"waterfall"	Rainy	31.64	0.9097	0.9550	0.6884	0.8915	0.0617	0.0
		TCL [41]	35.57	0.9578	0.9726	0.7138	0.9501	0.0242	2541.9
		DDN [31]	32.71	0.9517	0.9677	0.7162	0.9389	0.0407	35.0
		DIP [44]	36.95	0.9763	0.9765	0.8050	0.9721	0.0217	39.2
		FastDeRain	**39.88**	**0.9821**	**0.9891**	**0.8398**	**0.9783**	**0.0105**	**5.4**
	"highway"	Rainy	30.95	0.8592	0.9411	0.6711	0.7143	0.0974	0.0
		TCL [41]	34.63	0.9640	0.9728	0.6854	0.8863	0.0276	3837.2
		DDN [31]	29.59	0.9308	0.9521	0.6438	0.8093	0.0534	68.8
		MS-CSC [47]	37.47	0.9753	0.9818	0.8051	0.9209	0.0143	657.6
		DIP [44]	29.98	0.9668	0.9667	0.7799	0.9225	0.0698	31.7
		FastDeRain	**39.71**	**0.9874**	**0.9903**	**0.8692**	**0.9444**	**0.0078**	**5.2**
Case 2	"foreman"	Rainy	28.87	0.8991	0.9410	0.6654	0.7843	0.0922	0.0
		TCL [41]	30.77	0.9234	0.9486	0.5661	0.8159	0.0583	2003.1
		DDN [31]	33.21	0.9526	0.9671	0.6998	0.8564	0.0494	72.2
		DIP [44]	33.80	0.9650	0.9749	0.7133	**0.8943**	0.0379	13.9
		FastDeRain	**35.59**	**0.9694**	**0.9777**	**0.7272**	0.8905	**0.0306**	**2.5**
	"bus"	Rainy	26.16	0.8238	0.9300	0.5754	0.7822	0.1150	0.0
		TCL [41]	28.08	0.8668	0.9340	0.4967	0.8127	0.0839	4281.4
		DDN [31]	29.43	0.9171	0.9507	0.6097	0.8761	0.0644	47.2
		DIP [44]	29.79	0.9286	0.9582	0.6488	0.8927	0.0572	45.8
		FastDeRain	**31.99**	**0.9394**	**0.9644**	**0.6559**	**0.8998**	**0.0503**	**5.6**
	"waterfall"	Rainy	26.12	0.7827	0.8986	0.5303	0.7499	0.1096	0.0
		TCL [41]	29.14	0.8457	0.9210	0.4831	0.8142	0.0796	3089.2
		DDN [31]	30.45	0.8929	0.9370	0.5809	0.8685	0.0699	46.1
		DIP [44]	35.88	**0.9619**	0.9687	**0.7341**	**0.9542**	0.0253	37.4
		FastDeRain	**35.95**	0.9558	**0.9737**	0.7118	0.9459	**0.0223**	**5.6**
	"highway"	Rainy	28.23	0.8678	0.9359	0.6376	0.6815	0.1131	0.0
		TCL [41]	31.67	0.9294	0.9520	0.5506	0.7630	0.0494	2788.7
		DDN [31]	30.97	0.9380	0.9492	0.6055	0.7850	0.0609	44.9
		MS-CSC [47]	31.89	0.9678	0.9807	0.7859	0.8939	0.0144	277.1
		DIP [44]	29.68	0.9528	0.9590	0.6990	0.8499	0.0713	26.1
		FastDeRain	**39.94**	**0.9824**	**0.9873**	**0.8082**	**0.9113**	**0.0107**	**4.4**
Case 3	"foreman"	Rainy	23.76	0.9301	0.9631	0.7310	0.8289	0.0740	0.0
		TCL [41]	25.13	0.9321	0.9559	0.6138	0.8400	0.0582	1918.3
		DDN [31]	26.62	0.9586	0.9735	0.7398	0.8682	0.0487	115.9
		DIP [44]	26.29	0.9625	0.9750	0.7333	0.8889	0.0406	26.0
		FastDeRain	**27.27**	**0.9732**	**0.9822**	**0.7691**	**0.8965**	**0.0241**	**2.3**
	"bus"	Rainy	22.88	0.9101	0.9612	0.7371	0.8650	0.1067	0.0
		TCL [41]	25.85	0.8964	0.9485	0.5817	0.8383	0.0813	3944.0
		DDN [31]	25.73	0.9363	0.9640	0.7103	0.8903	0.0770	46.3
		DIP [44]	24.56	0.9306	0.9686	0.7026	0.8910	0.0699	47.1
		FastDeRain	**27.94**	**0.9611**	**0.9788**	**0.7653**	**0.9330**	**0.0447**	**5.5**
	"waterfall"	Rainy	22.35	0.9235	0.9587	0.7331	0.9016	0.0682	0.0
		TCL [41]	24.21	0.9226	0.9518	0.6238	0.9063	0.0463	2792.5
		DDN [31]	24.76	0.9417	0.9634	0.7201	0.9198	0.0566	46.0
		DIP [44]	23.44	0.9483	0.9713	0.7611	0.9349	0.0467	41.5
		FastDeRain	**26.03**	**0.9694**	**0.9822**	**0.8062**	**0.9643**	**0.0248**	**5.5**
	"highway"	Rainy	22.90	0.9212	0.9702	0.7666	0.7629	0.0683	0.0
		TCL [41]	24.12	0.9360	0.9659	0.6373	0.7909	0.0399	2433.8
		DDN [31]	25.06	0.9362	0.9566	0.6687	0.7883	0.0551	71.9
		MS-CSC [47]	24.20	0.9531	0.9797	0.7915	0.8898	0.0160	365.4
		DIP [44]	23.68	0.9464	0.9810	0.7956	0.8197	0.0417	40.8
		FastDeRain	**29.87**	**0.9721**	**0.9849**	**0.8125**	**0.8907**	**0.0130**	**4.9**

As observed in Table II, our method considerably outperformed the other four state-of-the-art methods in terms of all the selected quality assessment indexes. Notably, in many cases, the performances of the single-image-based deep learn-

[5]https://github.com/MinghanLi/MS-CSC-Rain-Streak-Removal

[6]http://www.2gei.com/video/effect/1_rain/

[7]http://trace.eas.asu.edu/yuv/

[8]http://www.changedetection.net

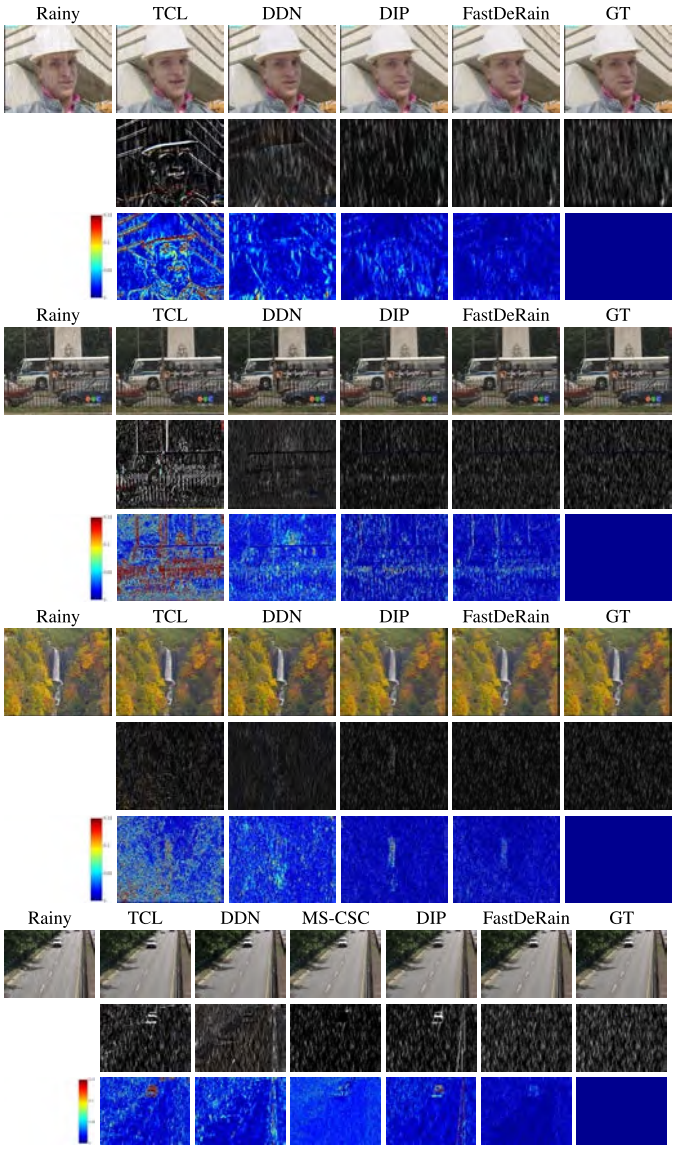


Fig. 5. The rainy frame, rain streaks removal results, extracted rain streaks and corresponding error images by different methods with synthetic rain streaks in **case 2**, respectively. The corresponding videos from top to bottom are the "'foreman", "bus", "waterfall" and "highway". From left to right are: the rainy data (or the color bar), results by TCL [41], DDN [31], DIP [44], (MS-CSC [47],) FastDeRain, and the ground truth (GT) clean video, respectively.

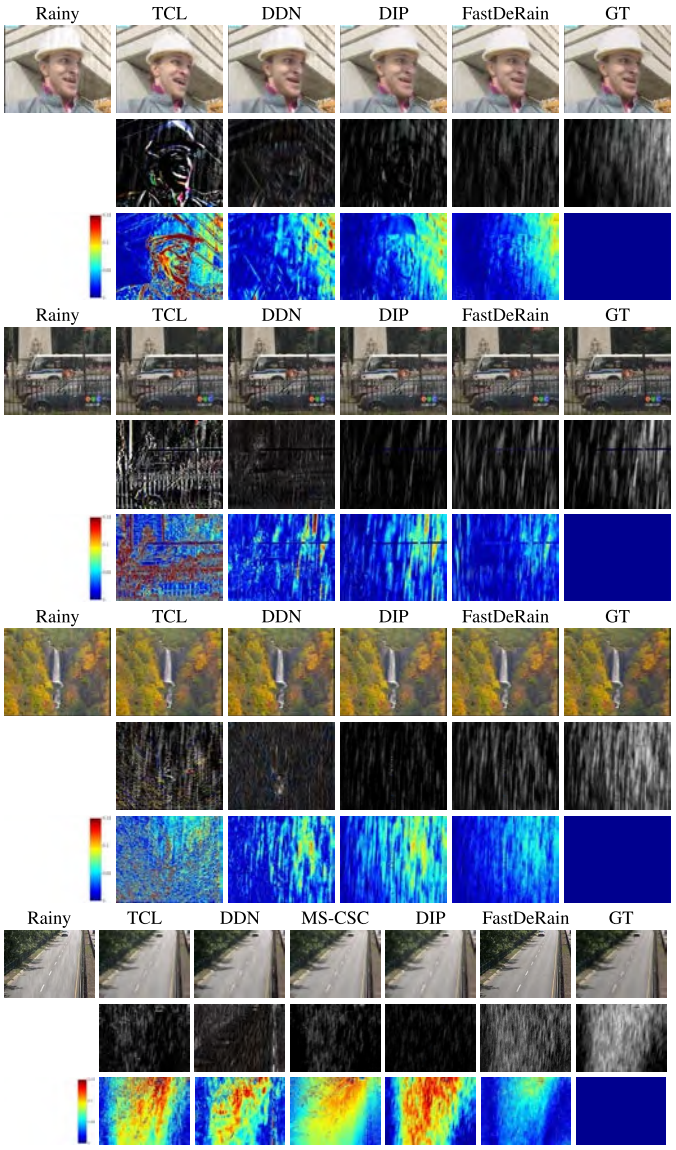


Fig. 6. The rainy frame, rain streaks removal results, extracted rain streaks and corresponding error images by different methods with synthetic rain streaks in **case 3**, respectively. The corresponding videos from top to bottom are the "'foreman", "bus", "waterfall" and "highway". From left to right are: the rainy data (or the color bar), results by TCL [41], DDN [31], DIP [44], (MS-CSC [47],) FastDeRain, and the ground truth (GT) clean video, respectively.

ing method DNN surpassed the those of the video-based method TCL. This is in agreement with the aforementioned rationality of considering comparisons with the single-image-based method.

The running time of the our FastDeRain is extremely low. In particular, our method took less than 10 seconds when dealing with all the synthetic data. The speed of DIP and DNN are comparably fast. After removing the nuclear norm term and avoiding the time consuming singular values decomposition, our algorithm, with closed-form solutions to its sub-problems and a time complexity of approximately $O(mnt\log(mnt))$ for an input video with a resolution of $m \times n$ and t frames, is expected to be efficient. In the meantime, the aforementioned implementation on the GPU device also largely accelerated our algorithm.

c) Visual comparisons: Fig. 4, 5 and 6 exhibit the results conducted on videos with synthetic rain streaks in case 1, case 2 and case 3, respectively. In Fig. 4, since the angles of rain streaks in case 1 increase with time, we display the frames at the beginning or end. Meanwhile, only one frame is exhibited in Fig. 5, Fig. 6 on account of that the rain streaks in every frame are of various directions.

In Fig. 4, all the methods removed almost all of the rain streaks and the proposed method maintained the best background. Many details in the background were incorrectly extracted to the rain streaks by DDN and TCL. It can be found in the 6-th row of Fig. 4, i.e., the error images of the results

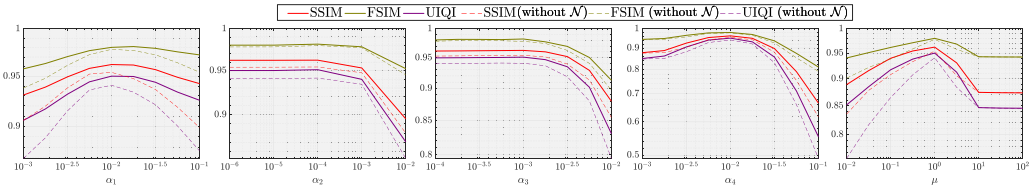


Fig. 7. The mean SSIM FSIM and UIQI values with respect to different values of α_1, α_2, α_3, α_4 and μ. The solid lines are corresponding to the results of FastDeRain while the dashed lines are related to the results obtained by our method without the $\mathcal{N}$ in Eq. (1).

on the video "bus", that little vertical patterns were mistakenly extracted as the rain streaks by the proposed method.

For the rain streaks in case 2, the denser rain streaks imply that it is more difficult than rain streaks in case 1. As we mentioned in Sec. III-B, the low-rank assumption is not reasonable for the videos with moving objects. The performance of DIP on the video "highway" was degraded. From Fig. 5, we can find that our method preserved the backgrounds well and other four methods erased the details of the backgrounds.

In Fig. 6, the proposed method removed most of the rain streaks and considerably preserves the background. Other methods tended to obtain over de-rain or under de-rain results. Considering the similarity of the extract rains streaks to the ground truth rain streaks, our FastDeRain held obvious advantages.

In summary, for these different types of synthetic data, our method can simultaneously remove almost all rain streaks while commendably preserving the details of the underlying clean videos.

d) Discussion of each component: There are four components in our model (2). To elucidate their distinct effects, we degrade our method by setting each α_i $(i = 1, 2, 3, 4)$ equal to 10^{-15}, respectively. These degraded methods and FastDeRain are tested on the video "waterfall" with synthetic rain streaks in case 1. We present the quantitative assessments in Fig. 9 and the visual results in Fig. 8.

From Fig. 9 and Fig. 8, we can conclude that all the four components contribute to the removal of rain streaks. Specifically, (a) when setting $\alpha_1 = 10^{-15}$, the rain streaks tend to be intermittent along the vertical direction; (b) the rain streaks are fatter when the sparsity term contributes little; (c) some rain streaks remain in the background when the horizontal smoothness of the background is not sufficiently enhanced; (d) the temporal continuity seems overwhelmingly important since that without this regularization term our method nearly failed.

e) Parameters: To examine the performance of the proposed FastDeRain with respect to different parameters, we conduct a series of experiments on the rainy data on synthetic video "waterfall" with the synthetic rain streaks in case 1 and the Gaussian noise with zero mean and standard deviation 0.02. In Fig. 7, a parameter analysis is presented and the SSIM FSIM and MUIQI are selected. Based on guidance from Fig. 7, our tuning strategy is as following: (1) set α_2 and α_3 as 10^{-4} and other α_is to 0.01, and $\mu = 1$, (2) tune α_1 and α_4 until the results are barely satisfactory,

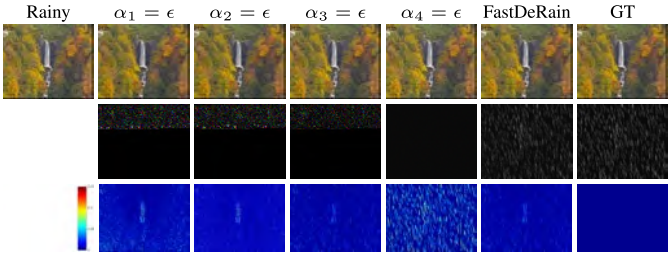


Fig. 8. The top row shows the 80th frame of the rainy video, the results by FastDeRain and its degraded versions, in which the α_is in Eq. (3) are set as $\epsilon = 10^{-15}$ in turn, and the ground truth (GT) clean video, respectively. The middle row presents the extracted rain streaks by FastDeRain and its degraded versions and the ground truth rain streaks, while the color bar and corresponding error images are exhibited in the bottom row.

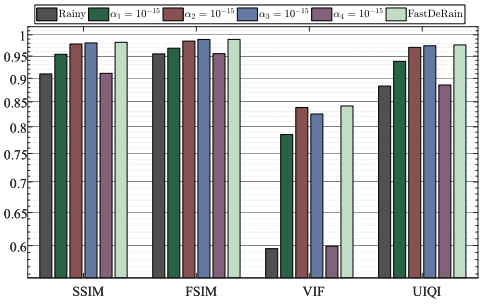


Fig. 9. The quantitative performances of the proposed method and its degraded versions, in which the α_is in Eq. (3) are set as 10^{-15} in turn.

(3) and then fix α_1 and α_4 and enlarge α_2 and α_3 to further improve the performance. The tuning principle is as follows: when some of the texture or detail of the clean video is extracted into the estimated rain streaks, we increase α_2 and α_1 or decrease α_4 and α_3, and we do the opposite when rain streaks remain in the estimated rain-free content. Our recommended set of candidate values for α_1 through α_4 is $\{0.0001, 0.0003, 0.001, 0.003, 0.01\}$. The Lagrange parameter μ is suggested to be 1. In practice, the time cost for the empirical tuning of the parameters is not much.

f) Discussion of the noise term $\mathcal{N}$ in Eq. (1): In this paper, the noise (or error) term ($\mathcal{N}$ in Eq. (1)) is taken into consideration in the observation model. To illustrate its effects, we conduct a series of experiments, in which the Gaussian noises of different standard deviations are respectively added to the video "waterfall" with synthetic rain streaks in case 1. The quantitative assessments of the results obtained by the proposed method with and without the noise (or error) term $\mathcal{N}$ taken into consideration (denoted as "w $\mathcal{N}$" and "w/o $\mathcal{N}$",

TABLE III

QUANTITATIVE COMPARISONS OF THE RAIN STREAK REMOVAL RESULTS OF THE PROPOSED FASTDERAIN WITH (W) AND WITHOUT(W/O) THE NOISE TERM TAKEN INTO CONSIDERATION ON SYNTHETIC VIDEO "WATERFALL" WITH THE SYNTHETIC RAIN STREAKS IN CASE 1. THE **BEST** QUANTITATIVE VALUES ARE IN **BOLDFACE**

σ	Method	PSNR	SSIM	FSIM	VIF	UIQI	GMSD
0	Rainy	31.63	0.9097	0.9550	0.5956	0.8834	0.0617
	w/o $\mathcal{N}$	**40.92**	**0.9869**	**0.9914**	**0.8629**	**0.9824**	**0.0099**
	w $\mathcal{N}$	40.52	0.9842	0.9900	0.8588	0.9787	0.0106
0.01	Rainy	31.04	0.9003	0.9516	0.5553	0.8611	0.0622
	w/o $\mathcal{N}$	37.95	0.9761	0.9868	0.8324	0.9684	0.0118
	w $\mathcal{N}$	**38.22**	**0.9764**	**0.9869**	**0.8373**	**0.9685**	**0.0111**
0.02	Rainy	29.64	0.8735	0.9422	0.4786	0.8042	0.0637
	w/o $\mathcal{N}$	34.86	0.9528	0.9764	0.7716	0.9386	0.0163
	w $\mathcal{N}$	**35.80**	**0.9622**	**0.9802**	**0.7983**	**0.9502**	**0.0132**
0.03	Rainy	27.99	0.8337	0.9286	0.4059	0.7317	0.0664
	w/o $\mathcal{N}$	33.22	0.9210	0.9666	0.7045	0.9017	0.0193
	w $\mathcal{N}$	**34.15**	**0.9387**	**0.9725**	**0.7329**	**0.9220**	**0.0162**
0.04	Rainy	26.41	0.7855	0.9125	0.3444	0.6566	0.0704
	w/o $\mathcal{N}$	31.63	0.8791	0.9540	0.6343	0.8558	0.0238
	w $\mathcal{N}$	**32.52**	**0.9038**	**0.9613**	**0.6593**	**0.8824**	**0.0211**

respectively) are reported in Table III. In addition, we also exhibit the effects of different parameters on the proposed method without $\mathcal{N}$ in Fig. 7.

From Table III, we can conclude our method without $\mathcal{N}$ would acquire a better result when the rainy video is free from the noise. However, when the video is simultaneously affected by the rain streaks and the noise, which is unavoidable in real data, our method with $\mathcal{N}$ got better results. Therefore, we adopt the term $\mathcal{N}$ in Eq. (3) which enhances the robustness of our method to the noise. Meanwhile, the solid lines and the dashed lines in Fig. 7 also demonstrate that taking the noise (or error) term $\mathcal{N}$ into account would contribute to the robustness of the proposed method to different parameters.

B. Real Data

In this section, four real-world rainy videos are chosen in this subsection. The first one (denoted as "wall") of size $288 \times 368 \times 3 \times 171$ is download from the CAVE dataset[9] and the second video[10](denoted as "yard") of size $512 \times 256 \times 3 \times 126$ was recorded by one of the authors on a rainy day in his backyard. The background of the video "wall" is consist of regular patterns while the background of the video "yard" is more complex. The third video is clipped from the well-known film "the Matrix". The scene in this clips changes fast so that it is more difficult to deal with this video. The last video of size $480 \times 640 \times 3 \times 108$ is denoted as "crossing"[11], and it was captured in the crossing with complex traffic conditions. In the mean time, to further illustrate the effects from noise term $\mathcal{N}$ in Eq. (1) in the experiments with real-world rainy data, we also exhibit the results obtained by our method without the noise (or error) term $\mathcal{N}$ taken into consideration (denoted as"w/o $\mathcal{N}$").

[9]http://www.cs.columbia.edu/CAVE/projects/camera_rain/

[10]https://github.com/TaiXiangJiang/FastDeRain/blob/master/yard.mp4

[11]https://github.com/hotndy/SPAC-SupplementaryMaterials/blob/master/Dataset_Testing_RealRain/ra4_Rain.rar

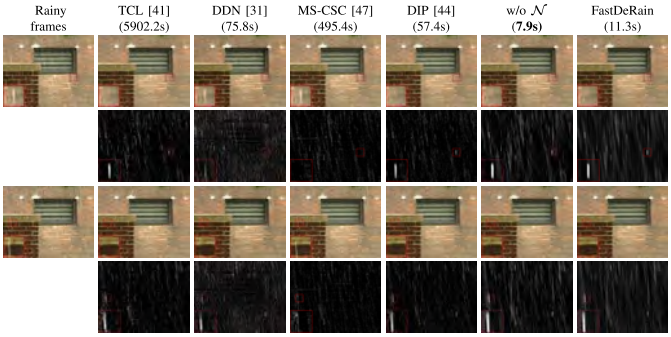


Fig. 10. Rain streak removal performance of different methods obtained on the video "wall". From top to bottom, two adjacent frames of the deraining results and corresponding extracted rain streaks are illustrated. From left to right are: the rainy data, results by different methods, and the ground truth.

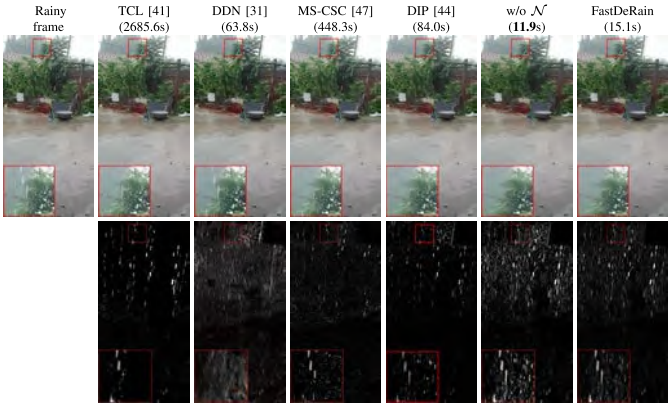


Fig. 11. Rain streak removal results on the video "yard". From left to right are frames of the rainy video, rain streaks removal results and corresponding extracted rain streaks by different methods, respectively. From left to right are: the rainy data, results by different methods, and the ground truth.

Fig. 10 shows two adjacent frames of the results obtained on the video "wall". There are many vertical line patterns in the background of this video. Thus, exhibiting two adjacent frames would further help to distinguish the rain streaks from the background. It can be found in the zoomed in red blocks that this rain streak with high brightness is not handled properly by DNN and MS-CSC. Our methods, including DIP and FastDeRain, remove almost all the rain streaks and preserves the background best compared with the results by other three methods. It can be found that the rain streaks acquired by our FastDeRain is more smooth along the vertical direction compared with the results obtained by our method without $\mathcal{N}$.

Since that there is little texture or structure similar to rain streaks in the video "yard", only one frame is exhibited in Fig. 11. DNN didn't distinguish most of the rain streaks, especially in the zoomed in red blocks. Although TCL and MS-CSC separated the majority of rain streaks, we could still observe remaining rain streaks in the zoomed in area. The deraining results got by DIP and FastDerain were similarly clean, and our FastDeRain without $\mathcal{N}$ incorrectly extracted some content of the background into the rain streaks.

In Fig. 12, two adjacent frames of the rainy video "the Matrix" and deraining results by different methods are shown.

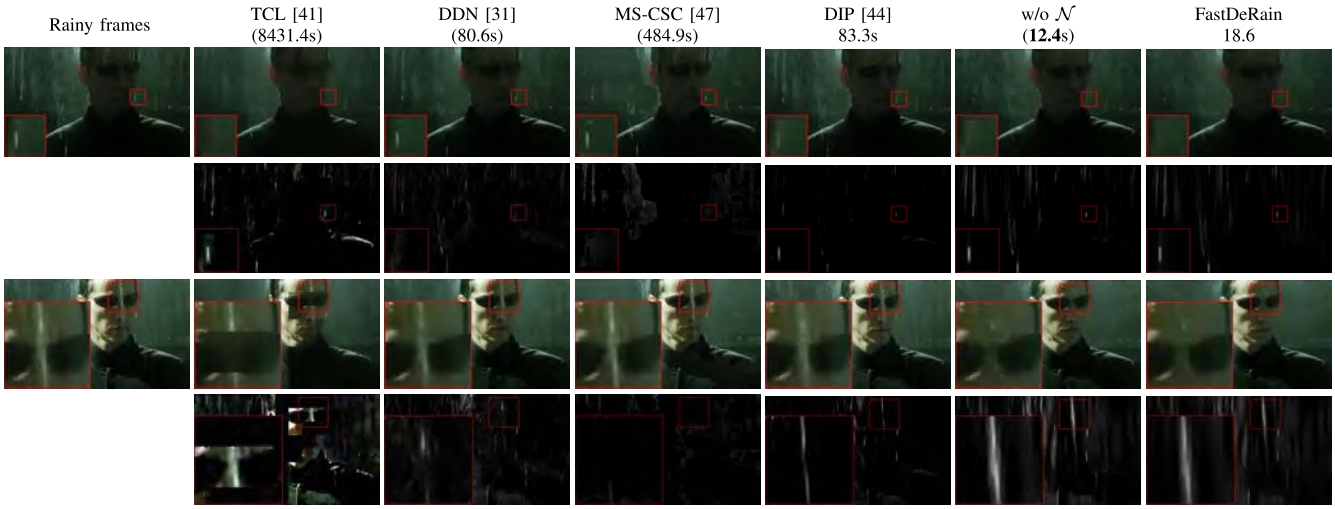


Fig. 12. Rain streak removal performance of different methods obtained on the clips of movie "the Matrix". From top to bottom, 2 adjacent frames of the rainy video/deraining results and corresponding extracted rain streaks are illustrated. From left to right are: the rainy data, results by different methods, and the ground truth.

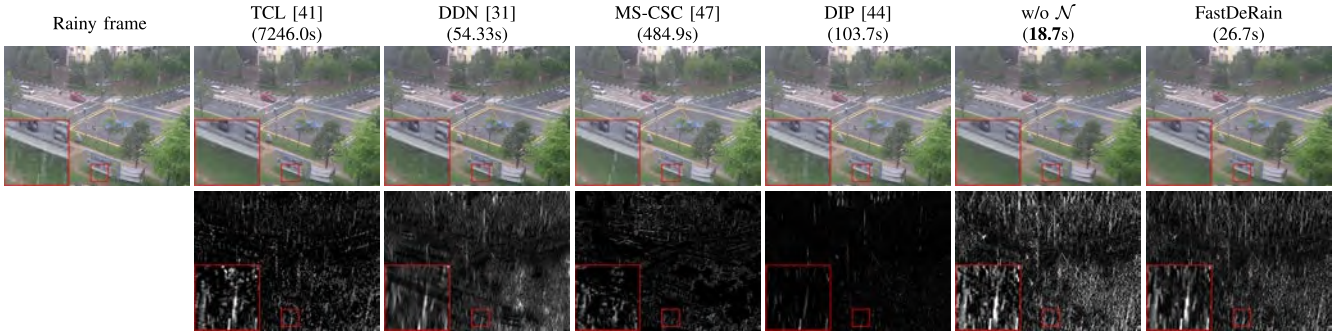


Fig. 13. Rain streak removal performance of different methods obtained on the video "crossing". From left to right are: the rainy data, deraining results or extracted rain streaks by different methods, and the ground truth.

The two adjacent rainy frames reveal the rapidly changing of the scene, particularly the luminance. For this video, DIP showed its limitation, remaining rain streaks in the deraining result. Once again, our FastDeRain obtained the best result, especially when dealing with the obvious rain streak on the face of Neo. Our FastDeRain comparatively outperformed its variation ersion without $\mathcal{N}$ in consideration of preservation of the face of Neo.

The results on the rainy video "crossing" are exhibited in Fig. 13. From the zoomed in areas in the first row, we can observe that TCL and our FastDeRain (with and without $\mathcal{N}$) acquired the most clean background while DNN MS-CSC and DIP left some rain streaks in the background more of less. The extracted rain streaks in the second row show that TCL extracted some the structure of the curb line into the rain streaks while DNN tended to remove all the textures with line pattern. The extracted rain streaks by the proposed FastDeRain were visually the best among all the results.

The scenarios in these four videos are of large differences. Our method obtains the best results, both in removing rain streaks and in retaining spatial details. In addition, the running time of our method is also obviously less than other methods, especially those video-based methods.

TABLE IV

QUANTITATIVE COMPARISONS OF THE RAIN STREAK REMOVAL RESULTS OF [41], [30], [46], [47] AND THE PROPOSED METHOD WITH THE SHIFT STRATEGY WHEN RAIN STREAKS ARE FAR AWAY FROM BEING VERTICAL. THE **BEST** QUANTITATIVE VALUES ARE IN **BOLDFACE**

Video: "waterfall" Angle: $15^\circ - 35^\circ$							
Method	PSNR	SSIM	FSIM	VIF	UIQI	GMSD	time (s)
Rainy	29.14	0.8612	0.9323	0.5111	0.8228	0.0754	—
TCL [41]	33.55	0.9336	0.9602	0.6362	0.9110	0.0363	2929.2
DDN [30]	32.10	0.9283	0.9589	0.5984	0.8993	0.0448	43.8
MS-CSC [47]	28.44	0.7593	0.8900	0.3876	0.6679	0.1154	264.3
DIP [44]	35.99	0.9619	0.9713	0.7683	0.9545	0.0228	43.0
FastDeRain	**38.01**	**0.9701**	**0.9838**	**0.8224**	**0.9597**	**0.0138**	**31.5**
Video: "highway" Angle: $35^\circ - 55^\circ$							
Method	PSNR	SSIM	FSIM	VIF	UIQI	GMSD	time (s)
Rainy	25.86	0.8325	0.9241	0.6572	0.6442	0.1044	0.0
TCL [41]	26.31	0.8948	0.9440	0.6562	0.6920	0.0666	1621.3
DDN [31]	28.73	0.8713	0.9236	0.5901	0.6718	0.0838	80.6
MS-CSC [47]	28.49	0.9605	0.9797	0.7944	0.8790	0.0167	379.8
DIP [44]	28.19	0.9205	0.9476	0.6779	0.7553	0.0735	35.3
FastDeRain	**37.02**	**0.9761**	**0.9836**	**0.8257**	**0.8887**	**0.0112**	**23.8**

C. Oblique Rain Streaks

In this subsection, we examine the performance of our method with the shift strategy and other four methods, when

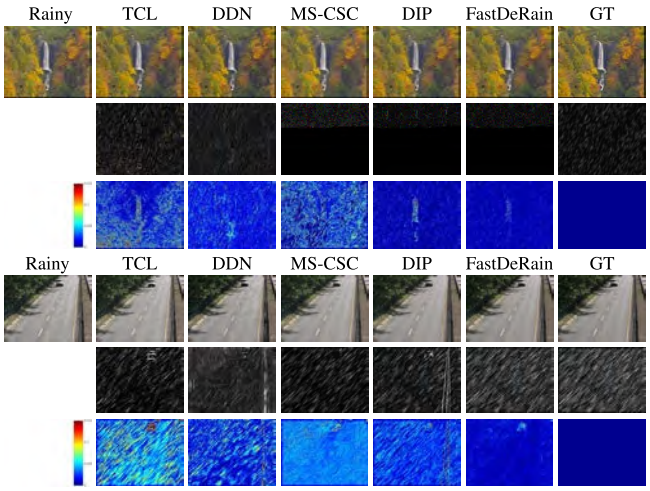


Fig. 14. From top to bottom are the rain streaks removal results, extracted rain streaks and corresponding error images by different methods on the video "highway1" (top 3 row) and "highway2" (bottom 3 row), respectively. From left to right are: the rainy data, results by TCL [41], DDN [31], MS-CSC [47], DIP [44] , and FastDeRain with the shift strategy, and the ground truth.

the rain streaks are far away from being vertical. We simulated two rainy videos: one is rain streaks with angles varying in $[15°, 35°]$ added to the video "waterfall" (captured by a dynamic camera); another one is rain streaks with angles varying in $[35°, 55°]$ added to the video "highway" (captured by a static camera). As shown in Table IV and Fig. 14, the shift strategy helped our method to obtains the best results when dealing with the oblique rain streaks. The superior of the proposed FastDeRain is obvious both quantitatively and visually.

V. Conclusion

We have proposed a novel video rain streaks removal approach: FastDeRain. The proposed method, based on directional gradient priors in combination with sparsity, outperforms a series of state-of-the-art methods both visually and quantitively. We attribute the outperforming of FastDeRain to our intensive analysis of the characteristic priors of rainy videos, clean videos and rain streaks. Besides, it notable that our method is markedly faster than the compared methods, even including a every fast single-image-based method. Our method is not without limitation. The natural rainy scenario is sometimes mixed with haze, and how to handle the residual rain artifacts remains an open problem. These issues will be addressed in the future.

Acknowledgment

The authors would like to express their sincere thanks to the editor and referees for giving us so many valuable comments and suggestions for revising this paper. They would like to thank Dr. Xueyang Fu and Dr. Minghan Li for their generous sharing of their codes.

References

[1] Y. Li, R. 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.

[2] Y.-F. Liu, D.-W. Jaw, S.-C. Huang, and J.-N. Hwang, "DesnowNet: Context-aware deep network for snow removal," *IEEE Trans. Image Process.*, vol. 27, no. 6, pp. 3064–3073, Jun. 2018.

[3] W. Ren *et al.*, "Deep video dehazing with semantic segmentation," *IEEE Trans. Image Process.*, to be published, doi: 10.1109/TIP.2018.2876178.

[4] B. Li *et al.*, "Benchmarking single-image dehazing and beyond," *IEEE Trans. Image Process.*, vol. 28, no. 1, pp. 492–505, Jan. 2019.

[5] 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.*, to be published, doi: 10.1109/TNNLS.2018.2862631.

[6] T. Bouwmans, "Traditional and recent approaches in background modeling for foreground detection: An overview," *Comput. Sci. Rev.*, vol. 11, pp. 31–66, May 2014.

[7] M. S. Shehata *et al.*, "Video-based automatic incident detection for smart roads: the outdoor environmental challenges regarding false alarms," *IEEE Trans. Intell. Transp. Syst.*, vol. 9, no. 2, pp. 349–360, Jun. 2008.

[8] 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.

[9] C. Ma, Z. Miao, X.-P. Zhang, and M. Li, "A saliency prior context model for real-time object tracking," *IEEE Trans. Multimedia*, vol. 19, no. 11, pp. 2415–2424, Nov. 2017.

[10] K. Garg and S. K. Nayar, "Vision and rain," *Int. J. Comput. Vis.*, vol. 75, no. 1, pp. 3–27, 2007.

[11] 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, Nov. 1998.

[12] 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.

[13] 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.

[14] 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. (ICCV)*, Dec. 2013, pp. 1968–1975.

[15] 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.

[16] D.-Y. Chen, C.-C. Chen, and L.-W. Kang, "Visual depth guided color image rain streaks removal using sparse coding," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 24, no. 8, pp. 1430–1455, Aug. 2014.

[17] 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.

[18] C.-H. Son and X.-P. Zhang, "Rain removal via shrinkage of sparse codes and learned rain dictionary," in *Proc. IEEE Int. Conf. Multimedia Expo Workshops (ICMEW)*, Jul. 2016, pp. 1–6.

[19] 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.

[20] 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.

[21] 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.*, Oct. 2017, pp. 2545–2553.

[22] B.-H. Chen, S.-C. Huang, and S.-Y. Kuo, "Error-optimized sparse representation for single image rain removal," *IEEE Trans. Ind. Electron.*, vol. 64, no. 8, pp. 6573–6581, Aug. 2017.

[23] 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.

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

[25] 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.

[26] 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.

[27] 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.

[28] 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. (ICCV)*, Dec. 2013, pp. 633–640.

[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] 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.

[31] 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. 3855–3863.

[32] 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 Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2018, pp. 2482–2491.

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

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

[35] 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.* Cham, Switzerland: Springer, 2018, pp. 262–277.

[36] R. Liu, Z. Jiang, L. Ma, X. Fan, H. Li, and Z. Luo, "Deep layer prior optimization for single image rain streaks removal," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, Apr. 2018, pp. 1408–1412.

[37] Z. Fan, H. Wu, X. Fu, Y. Huang, and X. Ding, "Residual-guide network for single image deraining," in *Proc. ACM Multimedia Conf.*, 2018, pp. 1751–1759.

[38] G. Li, X. He, W. Zhang, H. Chang, L. Dong, and L. Lin, "Non-locally enhanced encoder-decoder network for single image de-raining," in *Proc. ACM Multimedia Conf.*, 2018, pp. 1056–1064.

[39] K. Garg and S. K. Nayar, "Detection and removal of rain from videos," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2014, p. I.

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

[41] 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.

[42] 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.

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

[44] 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. 2818–2827.

[45] 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. 2838–2847.

[46] 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.

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

[48] 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 Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2018, pp. 1–10.

[49] J. Liu, W. Yang, S. Yang, and Z. Guo, "Erase or fill? Deep joint recurrent rain removal and reconstruction in videos," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2018, pp. 1–10.

[50] 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. 2018.

[51] S. Starik and M. Werman, "Simulation of rain in videos," in *Proc. IEEE Int. Conf. Comput. Vis. Texture Workshop (ICCV)*, vol. 2, 2003, pp. 1–6.

[52] X. Guo and Y. Ma, "Generalized tensor total variation minimization for visual data recovery?" in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 3603–3611.

[53] Y. Jiang, X. Jin, and Z. Wu, "Video inpainting based on joint gradient and noise minimization," in *Proc. Pacific Rim Conf. Multimedia*. Cham, Switzerland: Springer, 2016, pp. 407–417.

[54] D. Li, J. Leng, T. Huang, and G. Sun, "On sum and stability of g-frames in Hilbert spaces," *Linear Multilinear Algebra*, vol. 66, no. 8, pp. 1578–1592, 2018.

[55] Y. Chang, L. Yan, H. Fang, and H. Liu, "Simultaneous destriping and denoising for remote sensing images with unidirectional total variation and sparse representation," *IEEE Geosci. Remote Sens. Lett.*, vol. 11, no. 6, pp. 1051–1055, Jun. 2014.

[56] Y. Chang, L. Yan, T. Wu, and S. Zhong, "Remote sensing image stripe noise removal: From image decomposition perspective," *IEEE Trans. Geosci. Remote Sens.*, vol. 54, no. 12, pp. 7018–7031, Dec. 2016.

[57] H.-X. Dou, T.-Z. Huang, L.-J. Deng, X.-L. Zhao, and J. Huang, "Directional ℓ_0 sparse modeling for image stripe noise removal," *Remote Sens.*, vol. 10, no. 3, p. 361, 2018.

[58] R. Dian, L. Fang, and S. Li, "Hyperspectral image super-resolution via non-local sparse tensor factorization," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 3862–3871.

[59] T.-X. Jiang, T.-Z. Huang, X.-L. Zhao, T.-Y. Ji, and L.-J. Deng, "Matrix factorization for low-rank tensor completion using framelet prior," *Inf. Sci.*, vols. 436–437, pp. 403–417, Apr. 2018.

[60] S. Li, R. Dian, L. Fang, and J. M. Bioucas-Dias, "Fusing hyperspectral and multispectral images via coupled sparse tensor factorization," *IEEE Trans. Image Process.*, vol. 27, no. 8, pp. 4118–4130, Aug. 2018.

[61] T.-Y. Ji, N. Yokoya, X. X. Zhu, and T.-Z. Huang, "Nonlocal tensor completion for multitemporal remotely sensed images' inpainting," *IEEE Trans. Geosci. Remote Sens.*, vol. 56, no. 6, pp. 3047–3061, Jun. 2018.

[62] T. G. Kolda and B. W. Bader, "Tensor decompositions and applications," *SIAM Rev.*, vol. 51, no. 3, pp. 455–500, 2009.

[63] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Found. Trends Mach. Learn.*, vol. 3, no. 1, pp. 1–122, Jan. 2011.

[64] T.-X. Jiang, T.-Z. Huang, X.-L. Zhao, and L.-J. Deng. (2017). "A novel nonconvex approach to recover the low-tubal-rank tensor data: When t-SVD meets PSSV." [Online]. Available: https://arxiv.org/abs/1712.05870

[65] X.-L. Zhao, F. Wang, T.-Z. Huang, M. K. Ng, and R. J. Plemmons, "Deblurring and sparse unmixing for hyperspectral images," *IEEE Trans. Geosci. Remote Sens.*, vol. 51, no. 7, pp. 4045–4058, Jul. 2013.

[66] X.-L. Zhao, F. Wang, and M. K. Ng, "A new convex optimization model for multiplicative noise and blur removal," *SIAM J. Imag. Sci.*, vol. 7, no. 1, pp. 456–475, 2014.

[67] Y. Chen, T.-Z. Huang, X.-L. Zhao, and L.-J. Deng, "Hyperspectral image restoration using framelet-regularized low-rank nonnegative matrix factorization," *Appl. Math. Model.*, vol. 63, pp. 128–147, Nov. 2018.

[68] M. V. Afonso, J. M. Bioucas-Dias, and M. A. T. Figueiredo, "An augmented Lagrangian approach to the constrained optimization formulation of imaging inverse problems," *IEEE Trans. Image Process.*, vol. 20, no. 3, pp. 681–695, Mar. 2011.

[69] S. Ono, T. Miyata, and I. Yamada, "Cartoon-texture image decomposition using blockwise low-rank texture characterization," *IEEE Trans. Image Process.*, vol. 23, no. 3, pp. 1128–1142, Mar. 2014.

[70] H. W. Schreier, J. R. Braasch, and M. A. Sutton, "Systematic errors in digital image correlation caused by intensity interpolation," *Opt. Eng.*, vol. 39, no. 11, pp. 2915–2922, 2000.

[71] *GPU Computing*. Accessed: Feb. 16, 2018. [Online]. Available: https://www.mathworks.com/help/distcomp/run-built-in-functions-on-a-gpu.html

[72] *Adding Rain to a Photo With Photoshop*. Accessed: Feb. 16, 2018. [Online]. Available: https://www.photoshopessentials.com/photo-effects/rain/

[73] 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.

[74] 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.
[75] H. R. Sheikh and A. C. Bovik, "Image information and visual quality," *IEEE Trans. Image Process.*, vol. 15, no. 2, pp. 430–444, Feb. 2006.
[76] Z. Wang and A. C. Bovik, "A universal image quality index," *IEEE Signal Process. Lett.*, vol. 9, no. 3, pp. 81–84, Mar. 2002.
[77] 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.
[78] C. H. Bahnsen and T. B. Moeslund, "Rain removal in traffic surveillance: Does it matter?" *IEEE Trans. Intell. Transp. Syst.*, 2018, doi: 10.1109/TITS.2018.2872502.

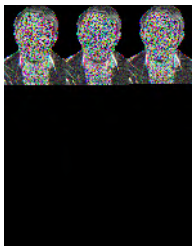
Tai-Xiang Jiang received the B.S. degrees in mathematics and applied mathematics from the Department of Mathematics, University of Electronic Science and Technology of China, Chengdu, China, in 2013, where he is currently pursuing the Ph.D. degree with the School of Mathematical Sciences. From 2017 to 2018, he a was a Co-Training Ph.D. Student with the Instituto Superior Técnico, Universidade de Lisboa, Lisbon, Portugal, supported by the China Scholarship Council.

His research interests include sparse and low-rank modeling, tensor decomposition, and deep learning. https://sites.google.com/view/taixiangjiang/.

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.

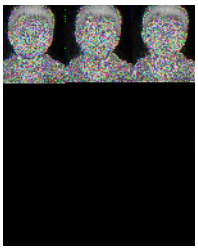
He is currently a Professor with the School of Mathematical Sciences, University of Electronic Science and Technology of China, Chengdu, China. His research interests include scientific computation and applications, numerical algorithms for image processing, numerical linear algebra, preconditioning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

Dr. Huang is an Editor of *The Scientific World Journal*, *Advances in Numerical Analysis*, the *Journal of Applied Mathematics*, the *Journal of Pure and Applied Mathematics: Advances in Applied Mathematics*, and the *Journal of Electronic Science and Technology*, China.



Xi-Le Zhao 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 main research interests are focused on the models and algorithms of high-dimensional image processing problems.



Liang-Jian Deng 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. He is currently a Lecturer with the School of Mathematical Sciences, UESTC. His research interests include image processing and computer vision.

Yao Wang received the Ph.D. degree in applied mathematics from Xi'an Jiaotong University, Xi'an, China, in 2014. He was a Visiting Student at the Georgia Institute of Technology from 2010 to 2011. He is currently an Assistant Professor with the School of Mathematics and Statistics, Xi'an Jiaotong University. His current research interests include statistical signal processing, high-dimensional data analysis, sparse recovery, and computational biology.