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 streak 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 results in the sparse distribution in the different directional gradient domain, respectively. 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 (TV) 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 (SALSA) 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 comprehensive quantitative performance measures, our approach outperforms other state-of-the-art methods especially on account of the running time.**

***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, 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 [2], such as event detection [3], object detection [4, 5], tracking [6], recognition [7], and scene analysis [8]. Therefore, the removal of rain streaks is an essential task, which has recently received considerable attention.

Numerous methods have been proposed to improve the visibility of images/videos captured with rain streak interference [9–39]. They can be split into two categories: methods based on multiple images/videos and single-image methods. 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.

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).

For the single-image de-raining task, Kang *et al*. [9] 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. However, learning HF image bases typically results in a loss of image details. To alleviate this problem, Sun *et al*. [10] exploited the structural similarity of the derived HF image bases. Nevertheless, the backgrounds estimated using their method still tend to be blurry. Chen *et al*. [11] considered the pattern of the rain streaks and the smoothness of the background, but the constraints in their objective function were not sufficiently strong. Discriminative sparse coding was adopted by Luo *et al.* [12]. The recent work by Li *et al*. [1, 13] utilized Gaussian mixture model (GMM) patch priors for rain streak removal, with the ability to account for rain streaks of different orientations and scales. Nonetheless, their method tends to yield over-smooth rain-free images, *i.e.*, the details of the no-rain image content are not well preserved. To cope with this issue, Zhu *et al*. [14] proposed a joint bi-layer optimization method progressively separate rain streaks from background details, in which the gradient statistics are analyzed. Meanwhile, in [17–19], the directional property of rain streaks received a lot of attention. Ren *et al.* [21] removed the rain streaks from the image recovery perspective. Wang *et al.* [20] took advantage the image decomposition and dictionary learning. The recently developed deep learning technique were also applied to the single image rain streaks removal task, and excellent results are obtained [22–27].

For the video rain streaks removal task, Garg *et al*. [28] firstly raised a video rain streaks removal method with comprehensive analysis of the visual effects of rain on an imaging system. Since then, many approaches have been proposed for the video rain streaks task and obtained good rain removing

* Corresponding author. Tel.: +86 28 61831016.

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, Sichuan 611731, P. R. China. Y. Wang is with the School of Mathematics and Statistics, Xian Jiaotong University, Xian 710049, P. R. China. E-mails: {taixiangjiang, yao.s.wang}@gmail.com, {tingzhuhuang, liangjian1987112}@126.com, xlzhao122003@163.com.

performance in videos with different rain circumstances many approaches have been proposed for the task of video rain streaks removal, and obtained the promising performances even with different rainy circumstances. For instance, Tripathi *et al*. [29] took the spatiotemporal properties into consideration. In [11], the similarity and repeatability of rain streaks are utilized, and a generalized low-rank appearance model was proposed. Additionally, comprehensive early existing video-based methods are reviewed in [30]. Whereafter, Kim *et al*. [31] considered the temporal correlation of rain streaks and the low-rank nature of clean videos. Santhaseelan *et al.* [32] detected and removed the rain streaks based on phase congruency features. You *et al.* [33] took the raindrops adhered to a windscreen or window glass into account. In [34], a novel tensor-based video rain streak removal approach was proposed considering the directional property. The rain streaks and the clean background were stochastically modeled as a mixture of Gaussians by Wei *et al.* [35] while Li *et al.* [36] utilized the multiscale convolutional sparse coding. Ren *et al.* [37] dealt the video desnowing and deraining task based on matrix decomposition. For the video rain streaks removal, the deep learning based methods also started to reveal their effectiveness [38, 39].

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 $\mathcal{O} = \mathcal{B}+\mathcal{R}+\mathcal{N}$, where $\mathcal{N}$ is the noise or error term. The goal of video rain streak removal methods are 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 complete utilization of the prior knowledge that is 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 [31, 35, 40]. 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.

From the spatial perspective, it has been widely recognized that natural images are largely piecewise smooth and their gradient fields are typically sparse [41, 42]. 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, 11]. However, the effects of the rain streaks to the vertical gradient and horizontal gradient are different, which is likewise noticed in [17–19]. Initially, for the sake of

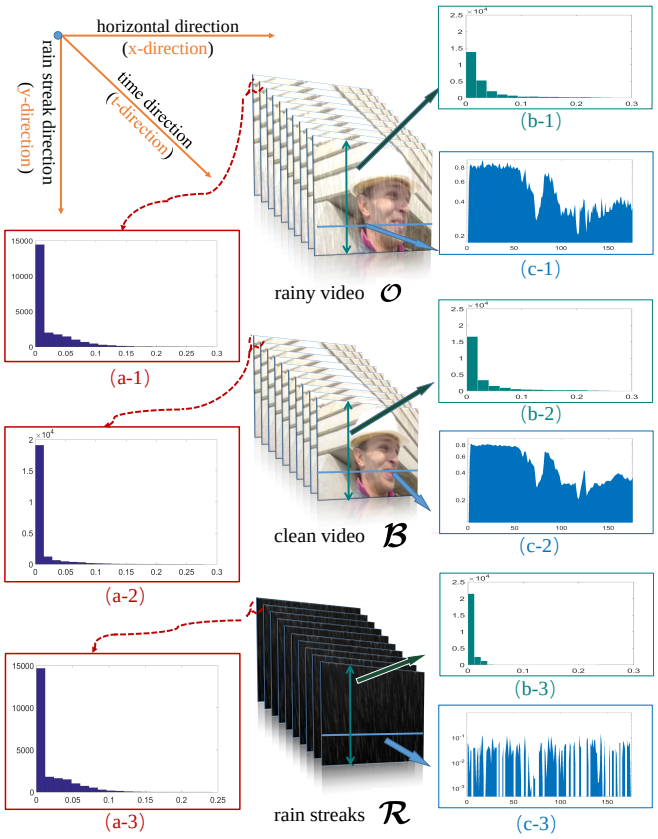

Fig. 2. From left to right: the histograms of temporal gradient of the rainy video (a-1), the clean video (a-2) and the isolated rain streaks (a-3), respectively; several example frames from the rainy video, the clean video and the isolated rain streaks; and the histograms of the vertical gradient (b-1,2,3)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

convenience, we assume that rain streaks are approximately vertical. The impact of the vertical rain streaks on the vertical gradient is limited. In Fig. 2, (b-1,2,3) reveal that the vertical gradient of rain streaks are much more sparse 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) operator and horizontal difference (or say horizontal unidirectional TV [43–45]) operator.

In a real rainfall-affected scene, without considering the wind, the raindrops generally fall from top to bottom. Meanwhile, when not very windy, the angles between rain streaks and the vertical direction is 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 propose two strategies, based on our automatical rain streaks' direction detection method.

The contribution of this paper mainly lies on 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 which 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 real rainy videos are conducted.

This work is an extension of the material published in [34]. 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 design an automatical rain streaks' direction detecting method and provide two strategies to deal with the rain streaks far from the vertical direction; 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) two recent state-of-the-art methods [23, 35] are brought into comparison.

The paper organized as follows. In Section II, some preliminary on the tensor notations is given. 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

TABLE I
TENSOR NOTATIONS

Notation	Explanation
$\mathcal{X}, \mathbf{X}, \mathbf{x}, x$	Tensor, matrix, vector, scalar.
$\mathbf{x}(:i_2 i_3 \cdots i_N)$	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 $\mathcal{X}$.

Following [46–48], we use lower-case letters for vectors, e.g., $\mathbf{a}$; upper-case letters for matrices, e.g., $\mathbf{A}$; 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.

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 $\langle \mathcal{X}, \mathcal{Y} \rangle := \sum_{i_1,i_2,\cdots,i_N} x_{i_1 i_2 \cdots i_N} \cdot y_{i_1 i_2 \cdots i_N}$. The corresponding norm (**Frobenius norm**) is then defined as $\|\mathcal{X}\|_F := \sqrt{\langle \mathcal{X}, \mathcal{X} \rangle}$.

Please refer to [48] 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. Therefore, when the rain streaks are dense, the ℓ_1 norm used can be viewed as a role to restrain the magnitude of the rain streaks. Meanwhile, we can tune the parameter of the sparsity term to handle a scene with heavy rain, as the rain streaks are always intrinsically sparser than the background clean video.

b) The horizontal direction: In Fig. 2, (c-1,2,3) show the pixel intensities along a fixed row for a 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). More zeros and smaller non-zero values 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 compared with rainy videos and rain streaks. Sub-figures (a-1,2,3), which present the hisgrams of the magnitudes of temporal directional gradient, illustrate that the a clean video's temporal gradient consist of more zero values and smaller non-zero values, whereas the those of a rainy video and rain streaks tend to include more and larger non-zero values. 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 [34] is discarded since that the low-rank assumption is not reasonable for the videos captured by dynamic cameras.

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 priors mentioned 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 succinctly formulated as follows:

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

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 [17, 34] An efficient algorithm is proposed in the following section 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 alternating direction method of multipliers (ADMM) [49], which has been proved an effective strategy for solving large scale optimization problems [18, 50–52]. More specifically, we use the SALSA [53].

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

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

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

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

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

a) $\mathcal{V}_i$ 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** $\boldsymbol{\mathcal{S}}_v(\cdot)$ is defined as

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

with

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

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

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

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

b) $\mathcal{B}$ and $\mathcal{R}$ sub-problems: : The $\mathcal{B}$ sub-problem is a least-squares problem:

$$\begin{aligned}\mathcal{B}^+=\arg\min_{\mathcal{O}\leq\mathcal{B}\leq\mathbf{0}}\ &\frac{1}{2}\|\mathcal{O}-\mathcal{B}-\mathcal{R}\|_F^2+\|\boldsymbol{\nabla}_2\mathcal{B}-\mathcal{V}_3-\mathcal{D}_3\|_F^2\\ &+\|\boldsymbol{\nabla}_t\mathcal{B}-\mathcal{V}_4-\mathcal{D}_4\|_F^2,\\ \mathcal{R}^+=\arg\min_{\mathcal{O}\leq\mathcal{R}\leq\mathbf{0}}\ &\frac{1}{2}\|\mathcal{O}-\mathcal{B}-\mathcal{R}\|_F^2+\|\boldsymbol{\nabla}_1\mathcal{R}-\mathcal{V}_1-\mathcal{D}_1\|_F^2\\ &+\|\mathcal{R}-\mathcal{V}_2-\mathcal{D}_2\|_F^2.\end{aligned}$$

Then, we have

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

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 $O(mnt\cdot\log(mnt))$.

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

$$\begin{cases} \mathcal{D}_1 = \mathcal{D}_1 + \boldsymbol{\nabla}_1\mathcal{R} - \mathcal{V}_1 \\ \mathcal{D}_2 = \mathcal{D}_2 + \mathcal{R} - \mathcal{V}_2 \\ \mathcal{D}_3 = \mathcal{D}_3 + \boldsymbol{\nabla}_2\mathcal{B} - \mathcal{V}_3 \\ \mathcal{D}_4 = \mathcal{D}_4 + \boldsymbol{\nabla}_t\mathcal{B} - \mathcal{V}_4 \end{cases} \tag{6}$$

Algorithm 1 FastDeRain[1]

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 $\mathcal{R}$

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 $O\left(mnt \log(mnt)\right)$.

E. Discussion of the oblique rain streaks

As we know that, in a real rainfall-affected scene, the rain streaks is not always vertical. Thus, the directional property we utilized in our model is a double-edged sword when dealing with digital videos. In this subsection, we design an automatically rain streaks' angle detection method, and based on it, we give two strategies to deal with rain streaks not vertical, namely, the rotation strategy and the shift strategy.

a) Rain streaks direction detection: Before stating our strategies, one important issue is how to automatically detect the direction of the rain streaks. Based on our analysis of the prior knowledge, it's not difficult to come up with a simple and effective method to detect the direction. In this subsection, we assume that the rain streaks are in the same direction and the angle between rain streaks and the vertical direction are denoted as θ. For a rainy video $\mathcal{O} \in \mathbb{R}^{m\times n\times t}$, our method consists of three steps:

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

2) Rotate each frame of $\mathcal{R}_0$ with $\theta_i = i^\circ$, and obtain $\mathcal{R}_0^{\theta_i}$ $(i = 0, 1, \cdots, t)$.

3) For each $\mathcal{R}_0^{\theta_i}$, denote $y_i = \|\nabla_1\mathcal{R}_0^{\theta_i}\|_1$, then the detected rain streaks angle $\hat{\theta} = \arg\min_{\theta_i} y_i$.

Fig. 3 shows an example of our detection method, where the rain streaks are simulated with angle 45° and the detection result (colored red) is exactly 45°. Actually, the y_is are very low when θ_i is close to 45°, echoing the discussion in III-B. For convenience, the range of $\hat{\theta}$ is restricted in $[0^\circ, 45^\circ]$ by simple flipping the frames upside down, or left to right.

[1]The demos of FastDeRain are available at https://github.com/uestctensorgroup/FastDeRain

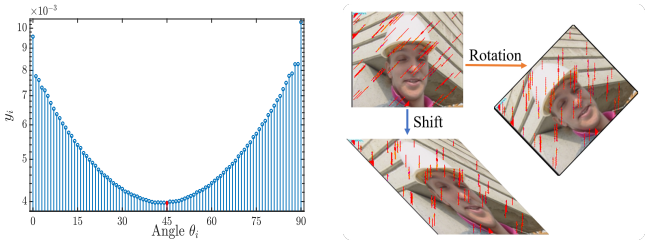

Fig. 3. Left: the magnitude of y_is with respect to θ_is. Right: an illustration of the rotation and shift, where the rain streaks are roughly labeled with the red color.

b) The shift strategy: When the angles are close to 45°, we can directly shift each row as shown in Fig. 3. After shifting, the rain streaks is close to being vertical, and we can apply the algorithm 1. Finally, the result would be shift back. It is notable that this shifting operation doesn't result in degradation of the video quality.

c) The rotation strategy: in another way, we can rotate the frames to make the rain streaks vertical. The rotation operation is shown in Fig. 3. Similarly, FastDeRain is applied to the rotated video, then the result would be rotated back. Actually, the rotation operation would unavoidably cause the video quality degradation, the degree of which would be fully analyzed in Sec. IV-C.

As mentioned before, the goal of rain streaks removal methods is to enhance the performance of the subsequent computer vision algorithms. Therefore, if the subsequent algorithms need data with high precision, we suggest the shift strategy. The flowchart of applying our FastDeRain with these two strategies are shown in Fig. 4.

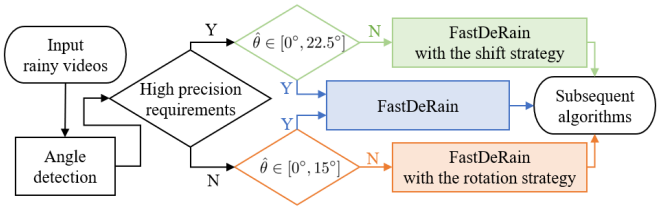

Fig. 4. The flowchart of the dealing with rainy videos with the rain streaks of different directions.

IV. Experimental Results

In this section, we evaluate the performance of the proposed algorithm on synthetic data and real rainy videos. All of the experimental results are uploaded in the uncompressed Audio Video Interleave (AVI) format on the website https://github.com/uestctensorgroup/FastDeRain. These AVI format videos provide a complete presentation of the excellent performance of our method.

a) Implementation details: Throughout our experiments, color videos with dimensions of $m \times n \times 3 \times t$ are transformed in 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

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

to the Y channel with the dimension of $m \times n \times t$. The rain streaks illustrated in this section are all added 0.5 for a 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 [54]. 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 can also be accelerated by the GPU device, from 38.13 seconds on the CPU to 24.17 seconds on the GPU, dealing with a video of size $240 \times 320 \times 3 \times 100$. 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 Fu *et al*.'s deep learning based single image rain streaks removal method[3] [23], Kim *et al*.'s method[4] based on temporal correlation and low-rankness [31], and Wei *et al*.'s method[5] using mixture of Gaussian [35]. In fact, Fu *et al*.'s method 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 Fu *et al*.'s deep learning based method 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 achieve the goal in one step. Meanwhile, as Starik *et al.* pointed out in [40] that the rain streaks can be assumed temporal independent, thus we can synthetic rain streaks for each frame using the synthetic method mentioned in many recently developed single image rain streaks removal approaches [9, 12, 23], *i.e.*, using the Photoshop software with the tutorial documents [55]. Another way to synthetic the rain streaks is proposed in [35], adding rain streaks taken by photographers under black background[6]. Referring to that tutorial and [35], we generate 3 types of rain streaks as follows:

Case 1 light rain streaks with fixed angles for each frame, and angles increase from -15° to 15° with time;

Case 2 heavy rain streaks with different angles varying in a range $[-15^\circ, 15^\circ]$;

Case 3 rain streaks' video taken by photographers under black background.

Four videos are selected as the clean background. The first two videos[7], named "highway1" and "foreman" with the size of $144 \times 176 \times 3 \times 160$, are captured by dynamic cameras, while the other two[8], named " highway2" and "truck" with the size of $240 \times 320 \times 3 \times 100$, are recorded by a static camera. The rain streaks in case 1 are added to videos " highway1" and "foreman". The synthetic rain streaks in case 2 are added to videos named " highway1", "foreman" and " highway2". In case 3, the rain streaks are added to two videos "highway2" and "truck".

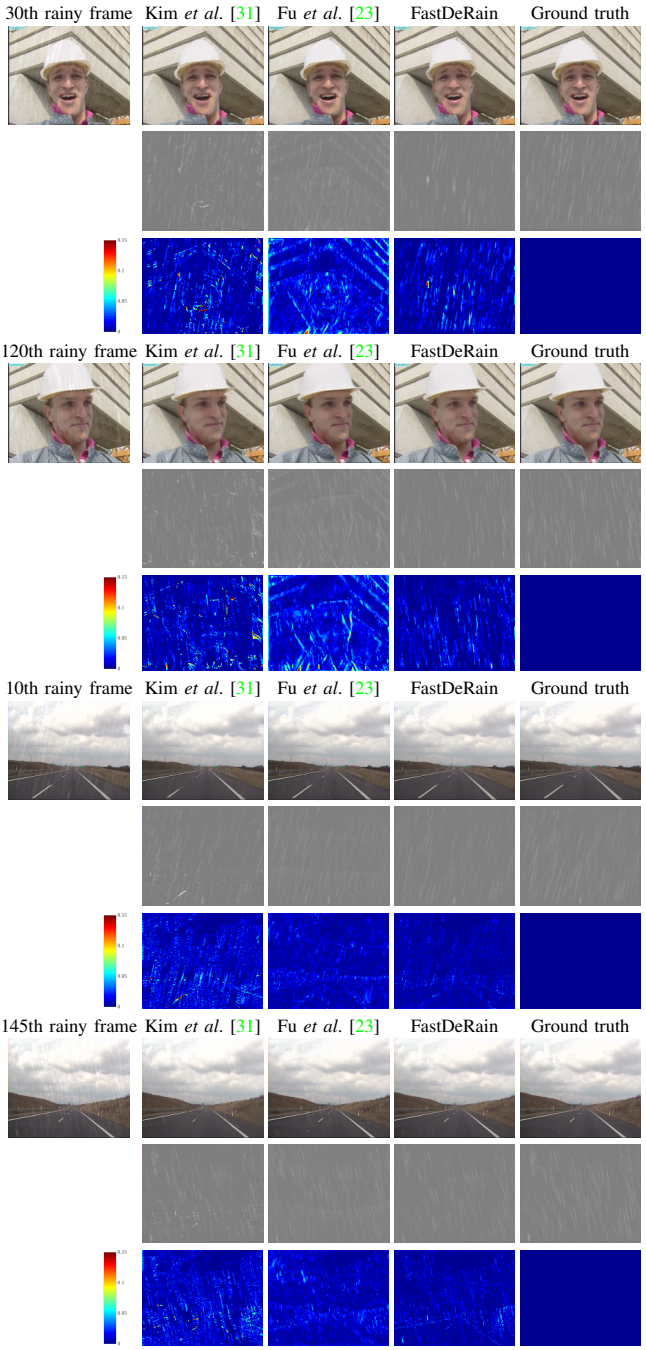

Fig. 5. The rain streaks removal results, extracted rain streaks and corresponding error images by different methods on the videos "foreman" (the top 6 rows) and "highway" (the bottom 6 rows) with synthetic rain streaks in case 1, respectively. From left to right are: the rainy data (or the color bar), results by [31], [23], FastDeRain, and the ground truth.

[3] http://smartdsp.xmu.edu.cn/xyfu.html
[4] http://mcl.korea.ac.kr/~jhkim/deraining/deraining_code_with_example.zip
[5] http://gr.xjtu.edu.cn/web/dymeng
[6] http://www.2gei.com/video/effect/1_rain/
[7] http://trace.eas.asu.edu/yuv/
[8] http://www.changedetection.net

TABLE II
QUANTITATIVE COMPARISONS OF THE RAIN STREAK REMOVAL RESULTS OF [31], [23], [35] AND THE PROPOSED METHOD ON SYNTHETIC VIDEOS. THE BEST QUANTITATIVE VALUES ARE IN BOLDFACE.

Rain streaks	Video	Method	PSNR	MSSIM	MFSIM	MVIF	MUIQI	MGMSD	Runing time (s)
Case 1	"foreman"	Rainy	34.6751	0.9753	0.9723	0.6787	0.8728	0.0524	—
		Kim *et al.* [31]	33.8645	0.9773	0.9715	0.6431	0.8946	0.0400	1696.47
		Fu *et al.* [23]	34.2572	0.9846	0.9804	0.7252	0.9172	0.0300	30.95
		Wei *et al.* [35]	20.4183	0.7835	0.7752	0.2568	0.4350	0.2138	544.01
		FastDeRain	**37.5777**	**0.9911**	**0.9867**	**0.7963**	**0.9422**	**0.0230**	**3.58**
	"highway1"	Rainy	35.0794	0.9540	0.9699	0.6345	0.8822	0.0530	—
		Kim *et al.* [31]	34.9299	0.9295	0.9822	0.6280	0.8225	0.0423	1409.62
		Fu *et al.* [23]	39.5537	0.9807	0.9890	0.7599	0.9406	0.0180	29.59
		Wei *et al.* [35]	27.2635	0.7589	0.9388	0.3743	0.5255	0.0739	249.55
		FastDeRain	**42.6449**	**0.9881**	**0.9916**	**0.8608**	**0.9674**	**0.0095**	**2.05**
Case 2	"foreman"	Rainy	28.8747	0.9441	0.9410	0.6323	0.7957	0.0922	—
		Kim *et al.* [31]	30.7556	0.9562	0.9486	0.5157	0.8233	0.0584	2625.81
		Fu *et al.* [23]	33.2129	0.9732	0.9671	0.6720	0.8668	0.0494	40.34
		FastDerain	**35.5898**	**0.9862**	**0.9777**	**0.7314**	**0.9108**	**0.0306**	**3.02**
	"highway1"	Rainy	29.4259	0.8992	0.9340	0.5097	0.7709	0.0935	—
		Kim *et al.* [31]	32.1216	0.9004	0.9697	0.4985	0.7530	0.0486	1636.22
		Fu *et al.* [23]	36.5116	0.9629	0.9786	0.6438	0.8864	0.0344	40.04
		FastDeRain	**37.2244**	**0.9729**	**0.9817**	**0.7207**	**0.9299**	**0.0220**	**2.87**
	"highway2"	Rainy	28.7322	0.8955	0.9427	0.5320	0.7003	0.1014	—
		Kim *et al.* [31]	31.7803	0.9440	0.9543	0.5481	0.7757	0.0472	2176.13
		Fu *et al.* [23]	31.2254	0.9461	0.9512	0.5861	0.7948	0.0569	24.17
		Wei *et al.* [35]	30.2120	0.9695	0.9819	0.7851	0.8982	0.0137	628.49
		FastDeRain	**37.9982**	**0.9850**	**0.9846**	**0.8422**	**0.9229**	**0.0119**	**5.11**
Case 3	"highway2"	Rainy	22.9023	0.9285	0.9702	0.7037	0.7678	0.0683	—
		Kim *et al.* [31]	24.1008	0.9407	0.9658	0.5565	0.7913	0.0551	2012.80
		Fu *et al.* [23]	25.0615	0.9393	0.9567	0.6279	0.7913	0.0401	29.46
		Wei *et al.* [35]	23.7811	0.9525	0.9805	0.7554	0.8903	0.0145	659.15
		FastDeRain	**30.1779**	**0.9756**	**0.9838**	**0.8191**	**0.8964**	**0.0135**	**5.72**
	"truck"	Rainy	28.0670	0.9321	0.9337	0.4827	0.8095	0.1199	—
		Kim *et al.* [31]	33.0121	0.9756	0.9721	0.6741	0.9252	0.0407	9295.26
		Fu *et al.* [23]	30.0826	0.9420	0.9477	0.4957	0.8343	0.0908	49.41
		Wei *et al.* [35]	30.3172	0.9773	0.9799	0.7118	0.9418	0.0372	672.90
		FastDeRain	**37.9642**	**0.9931**	**0.9906**	**0.7625**	**0.9754**	**0.0214**	**7.83**

Wei *et al.*'s method [35] is designed mainly for the videos with static camera, and directly applying their method on the video "highway1" and "foreman" results in poor performances (see the gray values Table II). Therefore, for a fair comparison, the compared methods include Fu *et al.*'s method [23] and Kim *et al.*'s method [31] when dealing with the synthetic rainy data generated from "highway1" and "foreman". For rainy data simulated by adding rain streaks to the video "highway2" and "truck", Wei *et al.*'s method 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) [56], the feature similarity (FSIM) [57], the visual information fidelity (VIF) [58], the universal image quality index (UQI) [59], and the gradient magnitude similarity deviation (GMSD, smaller is better) [60] of each frame are calculated. The PSNR, the corresponding mean SSIMs (MSSIM), mean FSIMs (MFSIM), mean VIFs (MVIF), mean UQIs (MUQI), mean MGSDs (MGMSD) and the running times for the synthetic data are reported in Table II, in which the best quantitative values are in boldface.

As observed in Table II, our method considerably outperforms the other three state-of-the-art methods in terms of all the selected quality assessment indexes. Notably, in many cases, the single-image-based deep learning method by Fu *et al.* [23] outperforms the video-based method by Kim *et al.* [31]. 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 requires only less than 10 seconds, although a tensor system might be expected to be computationally expensive. In fact, 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 accelerates a lot.

c) Visual comparisons: Fig. 5, 6 and 7 exhibit the results conducted on videos with synthetic rain streaks in case 1, case 2 and case 3, respectively. In Fig. 5, since the angles of rain streaks in case 1 increase with time, we display the frames at the beginning and the end, respectively. Meanwhile, only one frame is exhibited in Fig. 6, Fig. 7 on account of that the directions of rain streaks would not change markedly.

In Fig. 5, all the methods remove almost all of the rain streaks and the proposed method maintains the best background. Many details in the background are incorrectly ex-

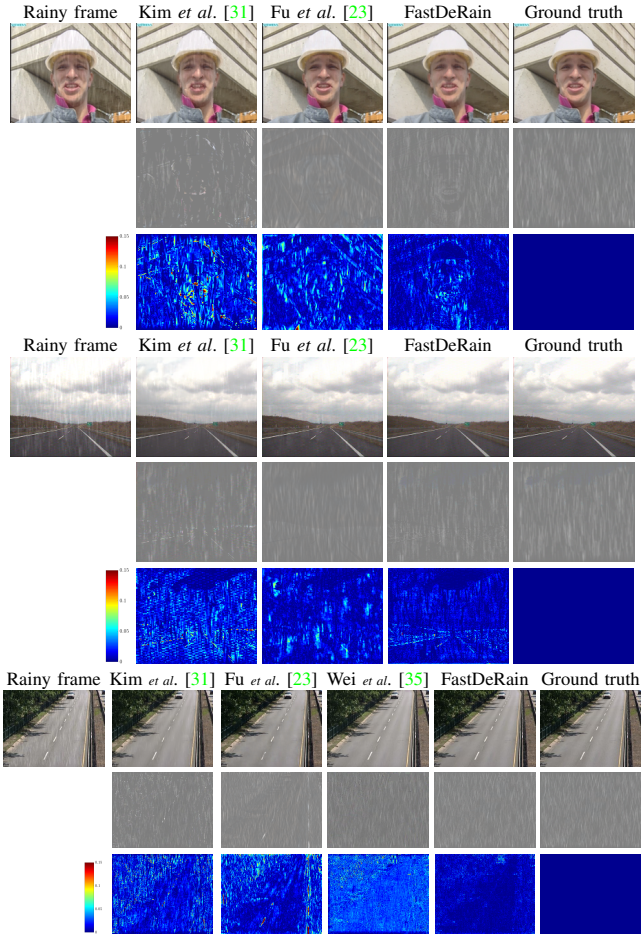

Fig. 6. The rain streaks removal results, extracted rain streaks and corresponding error images by different methods on the videos "foreman" (the top 3 rows), "highway1" (the middle 3 rows) and "highway2" (the bottom 3 rows) with synthetic rain streaks in case 2, respectively. From left to right are: the rainy data (or the color bar), results by [31], [23], ([35],) FastDeRain, and the ground truth.

tracted to the rain streaks by Fu *et al.*'s [23] and Kim *et al.*'s [31] methods. It can be found in the last row of Fig. 5 that little vertical patterns are improperly extracted as the rain streaks by the proposed method.

For the rain streaks in case 2, the heavier rain streaks imply that it is more difficult than dealing with case 1. From Fig. 6 we can conclude that all our method preserves the background well and other three methods erase the details of the background.

The visual results on synthetic rain streaks in case 3 are reported in Fig. 7. The proposed method and Wei *et al.*'s method [35] removes almost all the rain streaks and Kim *et al.*'s and Fu *et al.*'s methods leave some rain streaks in their deraining results. As for the extracted rain streaks, those by our method are the most similar to the ground truth rain streaks.

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

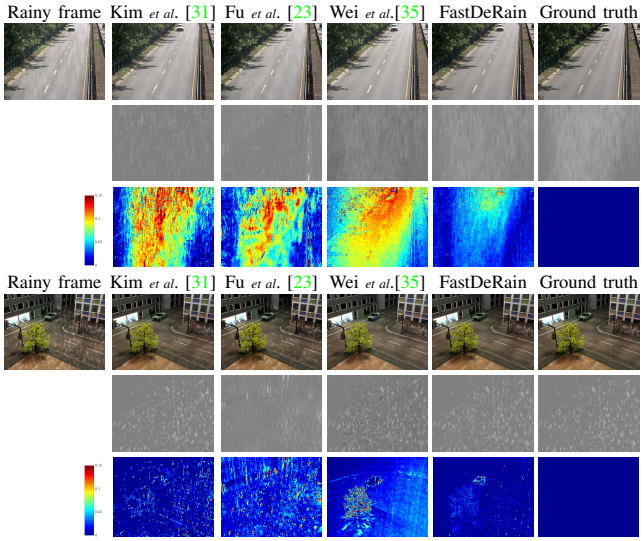

Fig. 7. The rain streaks removal results, extracted rain streaks and corresponding error images by different methods on the videos "highway2" (the top 3 row) and "truck" (the last 3 row) with synthetic rain streaks in case 3, respectively.

10^{-15}, respectively. These degraded methods and FastDeRain are tested on the video "highway1" with synthetic rain streaks in case 1. We present the quantitative assessments in Fig. 8 and the visual results in Fig. 9.

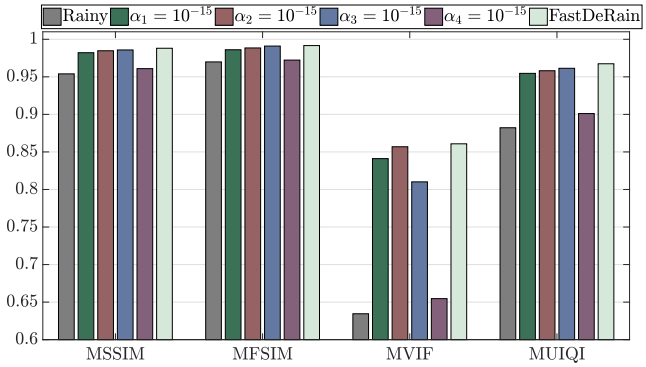

Fig. 8. The quantitative performances of the proposed method and its degraded versions, in which the α_is in Eq. (3) are set as 10^{-15} in turn.

From Fig. 8 and Fig. 9, 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 fails.

e) Parameters: In Fig. 10, a parameter analysis is presented. The MSSIM MFSIM MVIF and MUIQI are selected. It can be seen from Fig. 10 that the PSNR is slightly more sensitive to α_2 and α_4 than α_1, α_3 and β. Based on guidance from the parameter analysis, our tuning strategy is as following: (1) set all α_is to 0.01, and $\mu = 1$, (2) tune α_2 and α_4

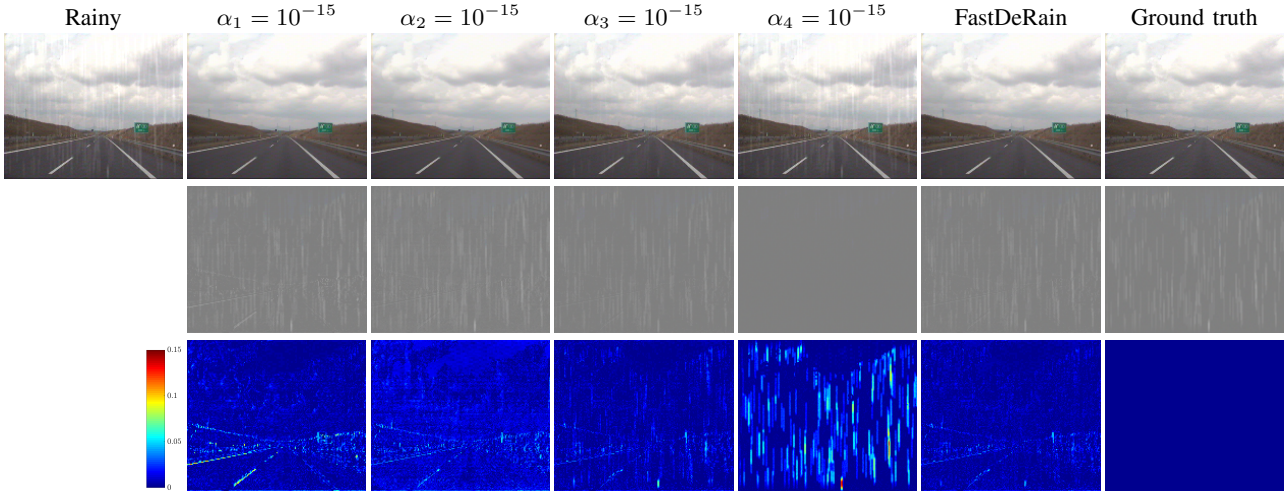

Fig. 9. The top row shows one frame of the rainy video, the results by FastDeRain and its degraded versions, in which the α_is in Eq. (3) are set as 10^{-15} in turn, and the clean video, respectively. The middle row presents the extracted rain streaks by FastDeRain and its degraded versions and the ground truth rain streaks, while the color bar and corresponding error images are exhibited in the bottom row.

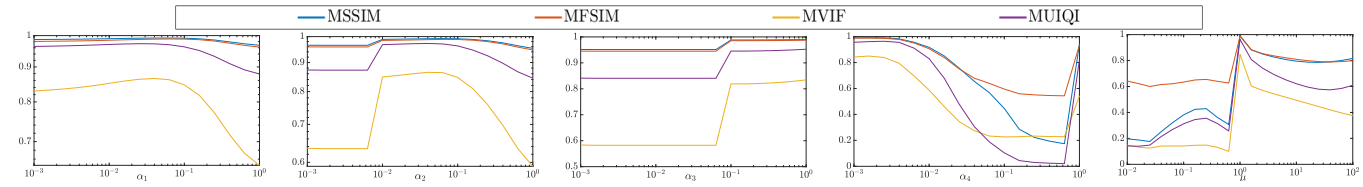

Fig. 10. The MSSIM MFSIM MVIF and MUIQI values with respect to different values of α_1, α_2, α_3, α_4 and μ.

until the results are barely satisfactory, (3) and then fix α_2 and α_4 and tune α_1 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 $\{0.0001, 0.0003, 0.001, 0.003, 0.01, 0.03\}$. The Lagrange parameter μ is suggested to be 1. In practice, the time cost for the empirical tuning of the parameters is not much.

B. Real data

Three videos are chosen for experiments on real data. The first one (denoted as "wall") is download from the CAVE dataset[9] and the second video[10] (denoted as "yard") 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 last 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.

Fig. 11 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 Fu *et al.*'s method and Wei *et al.*'s method. Our method removes almost all the rain streaks and preserves the background best compared with the results by other three methods.

Since there is little texture or structure similar to rain streaks in the video "yard", only one frame is exhibited in Fig. 12. Fu *et al.*'s method and Wei *et al.*'s method don't distinguish many rain streaks, especially in the zoomed in red blocks. As for the preservation of the background, our method and Fu *et al.*'s method perform better.

In Fig. 13, three adjacent frames of the rainy video "the Matrix" and deraining results by different methods are shown. The three adjacent rainy frames reveals the rapidly changing of the scene, particularly the luminance. Once again, our method removes almost all the rain streaks and preserves the background best. Interestingly, as shown in the fourth row of Fig. 13, the horizontal and vertical textures on the coat of Neo cause more or less troubles to all the methods.

The scenarios in these three 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 distinctly less than other methods, especially those two video-based method.

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

[10] Available at https://github.com/uestctensorgroup/FastDeRain/blob/master/yard.mp4

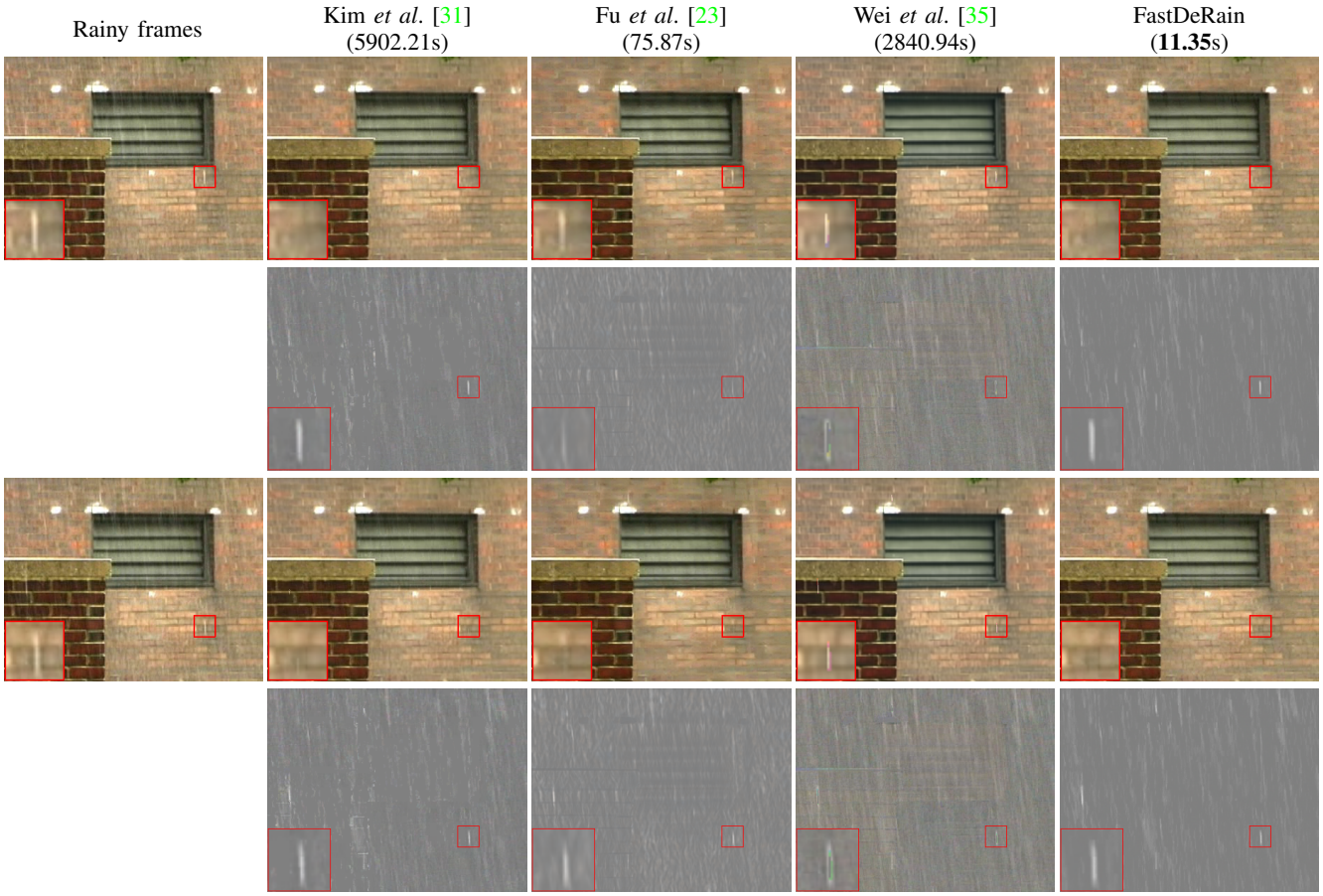

Fig. 11. Rain streak removal performance of different methods obtained on the video "wall". From top to bottom, two adjacent frames of the deraining results and corresponding extracted rain streaks are illustrated. From left to right are: the rainy data (or the color bar), results by [31], [23], [35], and FastDeRain.

C. Oblique rain streaks

In this subsection, we examine the performance of our method with shift strategy and rotation strategy and other three methods, when the rain streaks are far way from being vertical. We simulated two rainy videos: one is rain streaks with angles varying in $[40°, 50°]$ added to the video "highway2" (captured by a static camera); another one is rain streaks with angles varying in $[35°, 55°]$ added to the video "highway1" (captured by a dynamic camera).

As shown in Table III and Fig. 14, our method with the shift strategy obtains the best results. As for the rotation strategy, the two rotation operations (one to make the rain streaks vertical, and another one to rotate it back) would severely degrade the quality of the video. Specifically, the rotation mainly causes distortion for the fine and thin structures or textures (please see details in the last two columns of Fig. 14). Therefore, the superiority of our method with the rotation strategy on the quantitative assessment is not that much in contrast with the shift strategy. However, as shown in Fig. 14, except the distortion caused by the rotation operation, our method is still able to remove most of the rain streaks. Table III shows that our method with the rotation strategy can still achieve the second best (the best ones are carried out by our method with the shift strategy).

TABLE III
QUANTITATIVE COMPARISONS OF THE RAIN STREAK REMOVAL RESULTS OF [31], [23], [35] AND THE PROPOSED METHOD IN THE CASE THAT RAIN STREAKS ARE FAR AWAY FROM BEING VERTICAL. THE BEST QUANTITATIVE VALUES ARE IN BOLDFACE WHILE THE SECOND BEST ONES ARE UNDERLINED.

Video: "highway1"	Angle: $35° - 55°$						
Method	PSNR	MSSIM	MFSIM	MVIF	MUIQI	MGMSD	time (s)
Rainy	26.6317	0.8611	0.9011	0.4411	0.6969	0.1002	—
Rotation degradation*	31.5930	0.9458	0.9604	0.6123	0.8995	0.0500	
Kim *et al.* [31]	27.0355	0.8774	0.9247	0.4806	0.7024	0.0808	1793.96
Fu *et al.* [23]	29.6034	0.8955	0.9259	0.4913	0.7490	0.0787	32.84
Wei *et al.* [35]	22.7864	0.8571	0.9033	0.4521	0.6569	0.1280	370.42
FastDeRain (rotation)	<u>30.6694</u>	<u>0.9300</u>	<u>0.9477</u>	<u>0.5528</u>	<u>0.8523</u>	<u>0.0572</u>	<u>15.79</u>
FastDeRain (shift)	**36.5414**	**0.9713**	**0.9772**	**0.7835**	**0.9220**	**0.0277**	**9.55**
Video: "highway2"	Angle: $40° - 50°$						
Method	PSNR	MSSIM	MFSIM	MVIF	MUIQI	MGMSD	time (s)
Rainy	25.8620	0.8554	0.9241	0.5090	0.6511	0.1044	—
Rotation degradation*	38.5610	0.9916	0.9914	0.9593	0.9850	0.0044	
Kim *et al.* [31]	29.2639	0.9246	0.9449	0.5128	0.7207	0.0571	2220.83
Fu *et al.* [23]	31.2727	0.9502	0.9527	0.5840	0.7799	0.0565	41.46
Wei *et al.* [35]	27.1263	0.9422	<u>0.9607</u>	0.6626	0.8128	0.0346	179.20
FastDeRain (rotation)	<u>32.6591</u>	<u>0.9592</u>	0.9562	<u>0.7924</u>	<u>0.8402</u>	<u>0.0254</u>	<u>34.61</u>
FastDeRain (shift)	**37.0195**	**0.9810**	**0.9836**	**0.8382**	**0.8943**	**0.0125**	**18.44**

* The clean video with 2 rotating operations (one to make the rain perpendicular, and another one to rotate it back).

V. CONCLUSION

We have proposed a novel vdieo rain streaks removal approach: FastDeRain. The proposed method, based on di-

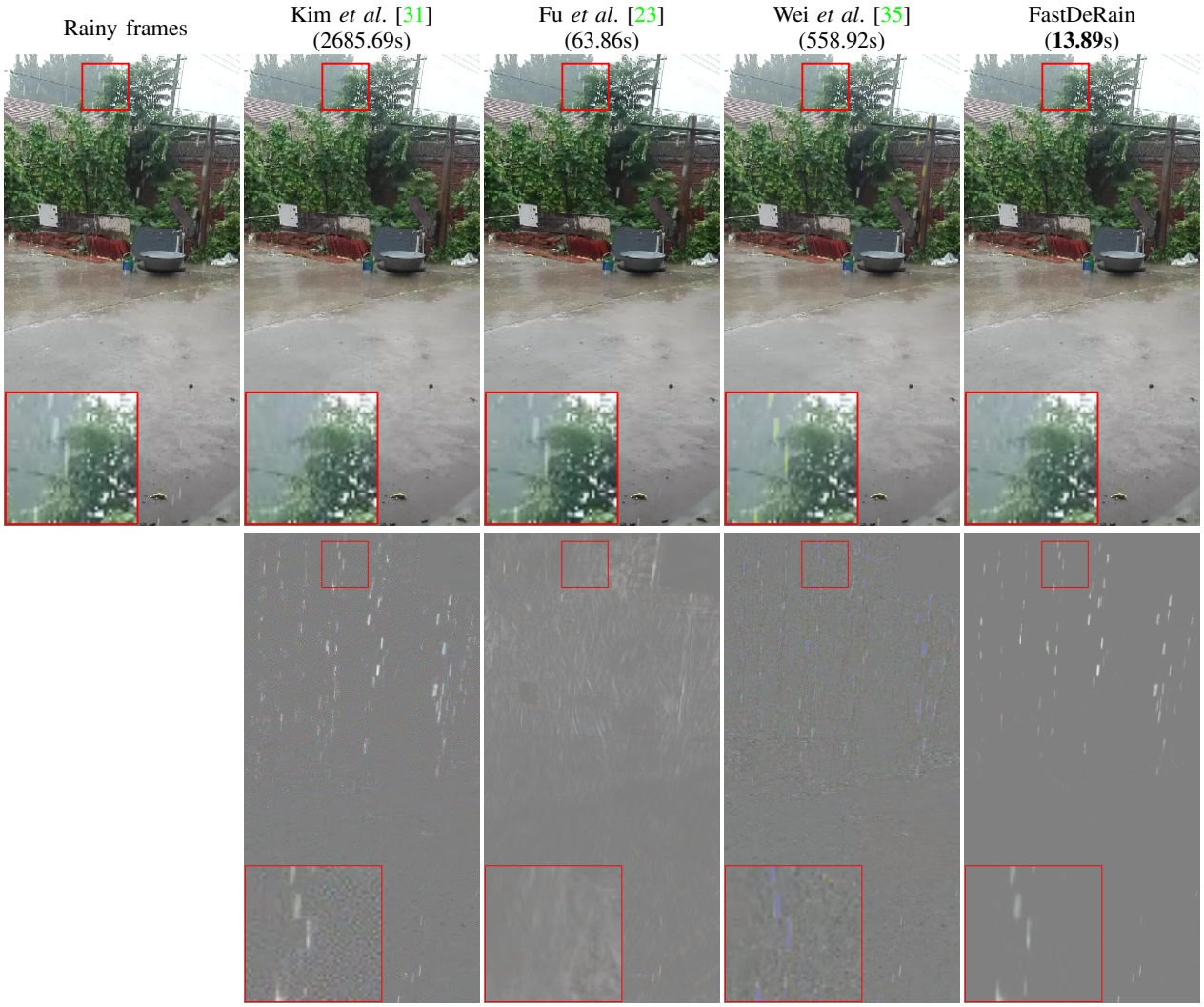

Fig. 12. Rain streak removal results on the video "wall". From left to right are frames of the rainy video, rain streaks removal results and corresponding extracted rain streaks by different methods, respectively. From left to right are: the rainy data, results by [31], [23], [35], and FastDeRain.

rectional 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 fast single-image-based method. Our method is not without limitations. The rotation strategy inevitably causes distortion for digital data. In addition, 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 future work.

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. The authors would like to thank Dr. Xueyang Fu and Dr. Wei Wei for their generous sharing of their codes. This research was supported by the National Natural Science Foundation of China (61772003, 61702083), and the Fundamental Research Funds for the Central Universities (ZYGX2016J132, ZYGX2016J129, ZYGX2016KYQD142).

References

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

[2] T. Bouwmans, "Traditional and recent approaches in background modeling for foreground detection: An overview," *Computer Science Review*, vol. 11, pp. 31–66, 2014.

[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*, vol. 9, no. 2, pp. 349–360, 2008.

[4] S. Maji, A. C. Berg, and J. Malik, "Classification using intersection kernel support vector machines is efficient," in *the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2008, pp. 1–8.

[5] O. L. Junior, D. Delgado, V. Gonçalves, and U. Nunes, "Trainable classifier-fusion schemes: An application to pedestrian detection," in *the IEEE International Conference on Intelligent Transportation Systems*, 2009, pp. 1–6.

[6] D. Comaniciu, V. Ramesh, and P. Meer, "Kernel-based object tracking," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 25, no. 5, pp. 564–577, 2003.

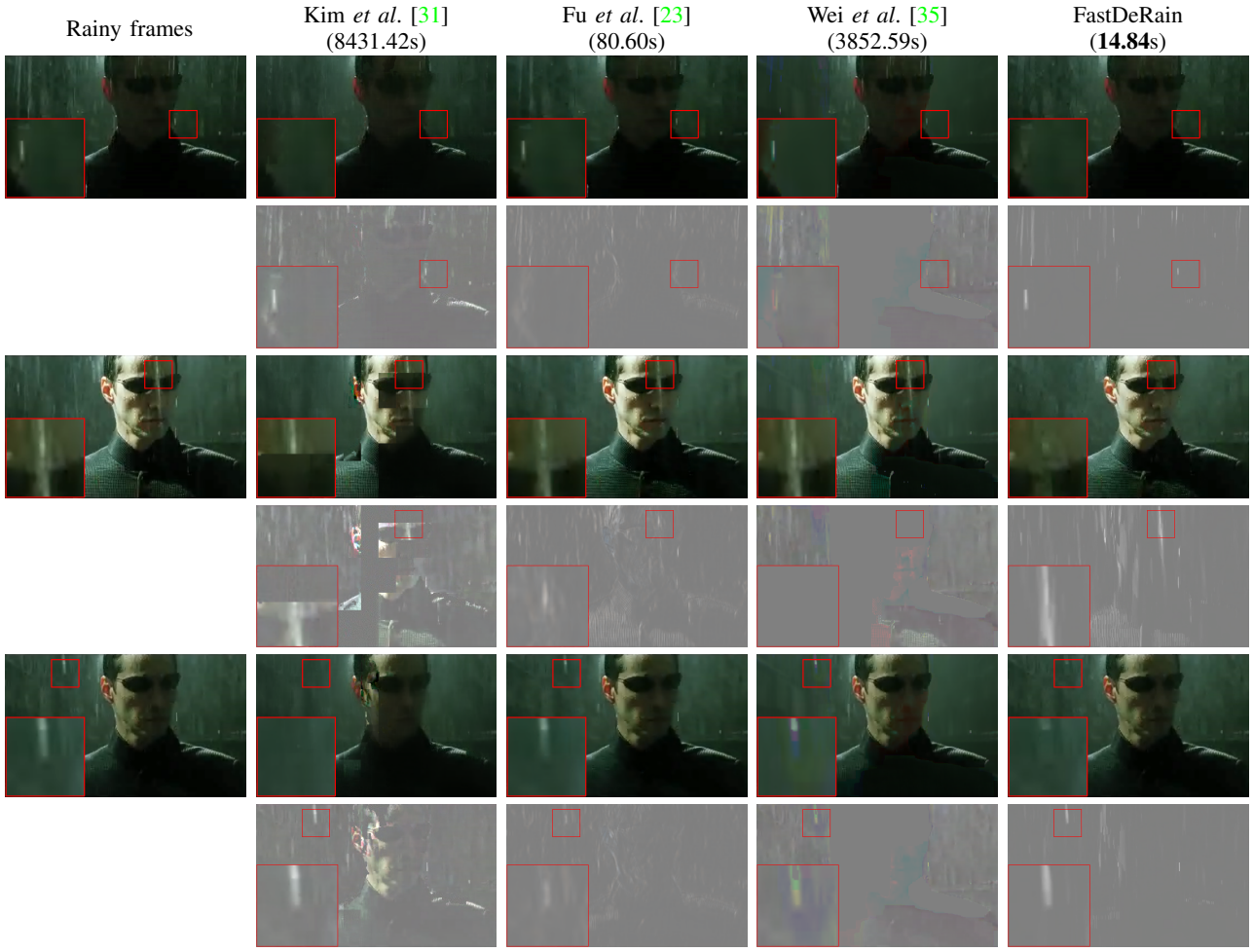

Fig. 13. Rain streak removal performance of different methods obtained on the clips of movie "the Matrix". From top to bottom, 3 adjacent frames of the rainy video/deraining results and corresponding extracted rain streaks are illustrated. From left to right are: the rainy data, results by [31], [23], [35], and FastDeRain.

[7] K. Garg and S. K. Nayar, "Vision and rain," *International Journal of Computer Vision*, vol. 75, no. 1, pp. 3–27, 2007.

[8] 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*, vol. 20, no. 11, pp. 1254–1259, 1998.

[9] 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*, vol. 21, no. 4, pp. 1742–1755, 2012.

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

[11] 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)*, 2013, pp. 1968–1975.

[12] 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)*, 2015, pp. 3397–3405.

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

[14] 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.

[15] B.-H. Chen, S.-C. Huang, and S.-Y. Kuo, "Error-optimized sparse representation for single image rain removal," *IEEE Transactions on Industrial Electronics*, vol. 64, no. 8, pp. 6573–6581, 2017.

[16] S. Gu, D. Meng, W. 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)*. IEEE, 2017, pp. 1717–1725.

[17] Y. Chang, L. Yan, and S. Zhong, "Transformed low-rank model for line pattern noise removal," in *the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 1726–1734.

[18] 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*, vol. 59, pp. 662–679, 2018.

[19] 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 Recognition*, vol. 79, pp. 303–317, 2018.

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

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

[22] 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 *the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

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

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

[25] R. Qian, R. T. Tan, W. Yang, J. Su, and J. Liu, "Attentive generative adversarial network for raindrop removal from a single image," *arXiv preprint arXiv:1711.10098*, 2017.

[26] S. Li, W. Ren, J. Zhang, J. Yu, and X. Guo, "Fast single image rain

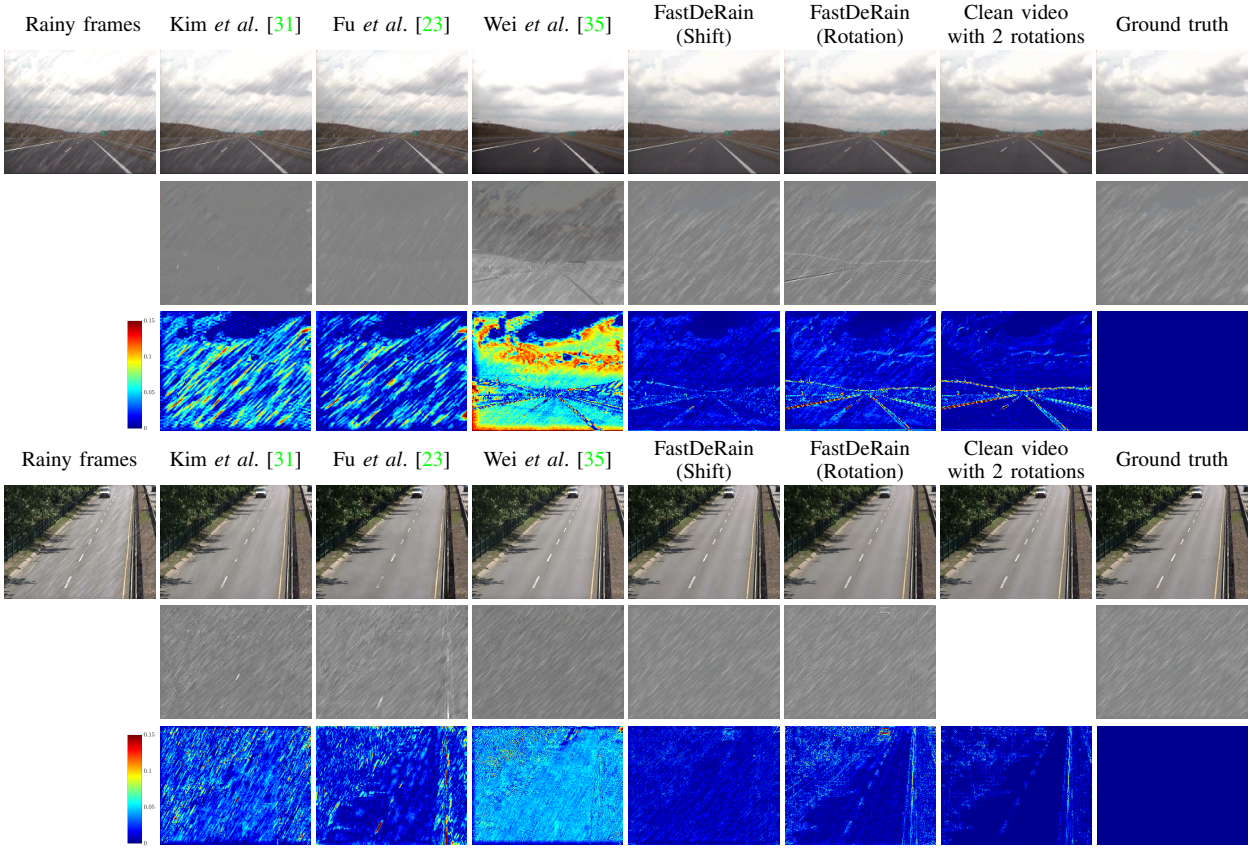

Fig. 14. From top to bottom are the rain streaks removal results, extracted rain streaks and corresponding error images by different methods on the video "highway1" (top 3 row) and "highway2" (bottom 3 row), respectively. From left to right are: the rainy data, results by [31], [23], [35], FastDeRain with shift strategy and FastDeRain with shift strategy, one frame of the clean video after two rotation operations, and the ground truth.

removal via a deep decomposition-composition network," *arXiv preprint arXiv:1804.02688*, 2018.

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

[28] K. Garg and S. K. Nayar, "Detection and removal of rain from videos," in *the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, vol. 1. IEEE, pp. I–528–I–535.

[29] A. Tripathi and S. Mukhopadhyay, "Video post processing: low-latency spatiotemporal approach for detection and removal of rain," *IET image processing*, vol. 6, no. 2, pp. 181–196, 2012.

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

[31] 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*, vol. 24, no. 9, pp. 2658–2670, 2015.

[32] V. Santhaseelan and V. K. Asari, "Utilizing local phase information to remove rain from video," *International Journal of Computer Vision*, vol. 112, no. 1, pp. 71–89, 2015.

[33] S. 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*, vol. 38, no. 9, pp. 1721–1733, 2016.

[34] 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)*, July 2017, pp. 4057–4066.

[35] 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 *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 2516–2525.

[36] M. Li, W. Wei, Q. Xie, Q. Zhao, j. Tao, and D. Meng, "Video rain streak removal by multiscale convolutional sparse coding," in *the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018, accepted.

[37] W. Ren, J. Tian, Z. Han, A. Chan, and Y. Tang, "Video desnowing and deraining based on matrix decomposition," in *the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 4210–4219.

[38] 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," *arXiv preprint arXiv:1803.10433*, 2018.

[39] S. Yang, S. Yang, W. Yang, and J. Liu, "Erase or fill? deep joint recurrent rain removal and reconstruction in videos," in *the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018, accepted.

[40] S. Starik and M. Werman, "Simulation of rain in videos," in *the IEEE International Conference on Computer Vision (ICCV) Texture Workshop*, vol. 2, 2003, pp. 406–409.

[41] X. Guo and Y. Ma, "Generalized tensor total variation minimization for visual data recovery," in *the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 3603–3611.

[42] Y. Jiang, X. Jin, and Z. Wu, "Video inpainting based on joint gradient and noise minimization," in *the Pacific Rim Conference on Multimedia*. Springer, 2016, pp. 407–417.

[43] 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 Geoscience and Remote Sensing Letters*, vol. 11, no. 6, pp. 1051–1055, 2014.

[44] Y. Chang, L. Yan, T. Wu, and S. Zhong, "Remote sensing image stripe noise removal: from image decomposition perspective," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 54, no. 12, pp. 7018–7031, 2016.

[45] H.-X. Dou, T.-Z. Huang, L.-J. Deng, X.-L. Zhao, and J. Huang, "Directional ?0 sparse modeling for image stripe noise removal," *Remote Sensing*, vol. 10, no. 3, p. 361, 2018.

[46] 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," *Information Sciences*, vol. 436, pp. 403–417, 2018.

[47] Y. Chang, L. Yan, H. Fang, S. Zhong, and Z. Zhang, "Weighted low-rank tensor recovery for hyperspectral image restoration," *arXiv preprint*

arXiv:1709.00192, 2017.

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

[49] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Foundations and Trends® in Machine Learning*, vol. 3, no. 1, pp. 1–122, 2011.

[50] T.-X. Jiang, T.-Z. Huang, X.-L. Zhao, and L.-J. Deng, "A novel nonconvex approach to recover the low-tubal-rank tensor data: when t-svd meets pssv," *arXiv preprint arXiv:1712.05870*, 2017.

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

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

[53] M. V. Afonso, J. M. Bioucas-Dias, and M. A. Figueiredo, "An augmented lagrangian approach to the constrained optimization formulation of imaging inverse problems," *IEEE Transactions on Image Processing*, vol. 20, no. 3, pp. 681–695, 2011.

[54] "GPU computing," https://www.mathworks.com/help/distcomp/run-built-in-functions-on-a-gpu.html.

[55] "Adding rain to a photo with photoshop," https://www.photoshopessentials.com/photo-effects/rain/.

[56] 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*, vol. 13, no. 4, pp. 600–612, 2004.

[57] L. Zhang, L. Zhang, X. Mou, and D. Zhang, "Fsim: A feature similarity index for image quality assessment," *IEEE transactions on Image Processing*, vol. 20, no. 8, pp. 2378–2386, 2011.

[58] H. R. Sheikh and A. C. Bovik, "Image information and visual quality," *IEEE Transactions on image processing*, vol. 15, no. 2, pp. 430–444, 2006.

[59] H. Prashanth, H. Shashidhara, and B. M. KN, "Image scaling comparison using universal image quality index," in *the International Conference onAdvances in Computing, Control, & Telecommunication Technologies,*. IEEE, 2009, pp. 859–863.

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