Contents lists available at ScienceDirect

Journal of Computational and Applied Mathematics

journal homepage: www.elsevier.com/locate/cam

Unsupervised video rain streaks removal with deep foreground–background modeling

Jun-Hao Zhuang [a,1], Yi-Si Luo [b,1], Xi-Le Zhao [c,*], Tai-Xiang Jiang [d], Yi Chang [e], Jun Liu [f]

[a] *Yingcai Honors College, University of Electronic Science and Technology of China, Chengdu, Sichuan, 610000, PR China*
[b] *School of Mathematics and Statistics, Xi'an Jiaotong University, Xi'an, Shaanxi, 710000, PR China*
[c] *School of Mathematical Sciences, University of Electronic Science and Technology of China, Chengdu, Sichuan, 610000, PR China*
[d] *School of Economic Information Engineering, Southwestern University of Finance and Economics, Chengdu, Sichuan, 610000, PR China*
[e] *School of Artificial Intelligence and Automation, Huazhong University of Science and Technology, Wuhan, Hubei, 430000, PR China*
[f] *School of Mathematics and Statistics, Northeast Normal University, Changchun, Jilin, 130000, PR China*

A R T I C L E   I N F O

Article history:
Received 18 November 2022
Received in revised form 20 April 2023

Keywords:
Video processing
Rain streaks removal
Inverse problem
Unsupervised
Deep convolutional neural network

A B S T R A C T

Outdoor video rain streaks removal is an important inverse problem in video processing that benefits subsequent applications. Traditional methods utilize prior information with interpretable domain knowledge while they are not tenable to capture complex structures of real-world videos. Deep learning methods learn a deraining mapping with a large model capacity brought by deep neural networks and their performances highly depend on the volume and diversity of training data. To address the challenging video deraining problem, we suggest an unsupervised video rain streaks removal method by solely using the observed rainy video. For the complex clean video, inspired by the classical foreground–background decomposition, we employ a deep convolutional neural network to capture the moving foreground and a disentangled deep spatial–temporal network with an affine operator to capture the underlying low-rank structure of the dynamic background. The foreground and background components are well balanced by a learnable probability mask. For structured rain streaks, we introduce a learnable total variation regularization whose parameters (i.e., rain directions) can be unsupervisedly learned. The deep modeling of the complex clean video and the simple yet effective modeling of structured rain streaks under the physical interpretable decomposition framework, which benefit each other in nature, are organically integrated to boost the deraining performance. Extensive experiments on synthetic and real-world rainy videos demonstrate the superiority of our method over state-of-the-art traditional and deep learning-based video deraining methods.

1. Introduction

Videos recorded in the rainy weather usually undergo rain streaks, which severely affect the visual quality and subsequent applications, e.g., classification [1], detection [2], and segmentation [3]. Therefore, it is of great importance to remove the rain streaks from the rainy videos [4], which is an important inverse problem in video processing (see Fig. 1).

* Corresponding author.
E-mail address: xlzhao122003@163.com (X.-L. Zhao).
[1] Contribute equally to this work.

https://doi.org/10.1016/j.cam.2023.115431
0377-0427/© 2023 Elsevier B.V. All rights reserved.

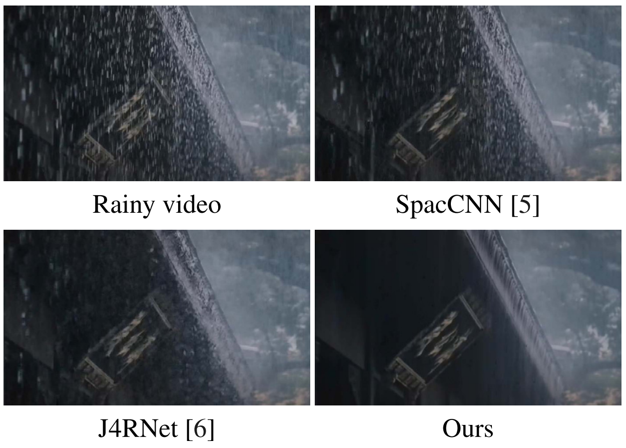

Fig. 1. The visual results by different video deraining methods on a real rainy video frame. Our unsupervised method outperforms state-of-the-art supervised learning methods [5,6].

Many traditional methods have been proposed for video deraining [7–10]. These methods utilize prior information of rain streaks [11–17] and clean videos [18–21] to form optimization models, which are solved by algorithms such as expectation–maximization [15], alternating minimization [14], or alternating direction method of multipliers (ADMM) [19, 21]. These traditional methods are suitable for both synthetic rainy videos and real-world rainy videos since they are based on hand-crafted prior knowledge, which does not depend on training data.

Inspired by the success of deep learning for inverse problems in imaging [22–24], many deep learning methods were recently proposed for video deraining. These methods design deep neural networks for end-to-end deraining with abundant training data [5,6,25–33]. The deep neural networks hold high representation abilities and thus can learn complex deraining mapping.

Although these traditional and deep learning methods achieve promising performance, they still face the following challenges for video deraining.

(1) The clean video has complex structures, e.g., moving foreground and dynamic background. For traditional methods, they are hard to fully depict the complex scenarios due to insufficient representation abilities. For deep learning methods, they are hard to collect synthetic training data containing all these complex scenes. Thus, deep leaning methods suffer from domain shift between training data and real-world complex rain scenarios and their performances would degrade on real-world data. How to robustly and appropriately model the complex clean video is one important challenge.
(2) Rain streaks, which are usually in line patterns, are more structured as compared with clean videos. How to elegantly model the structures of rain streaks, e.g., the rain directions, is another crucial aspect for video deraining.

To tackle these challenges, we propose an unsupervised video deraining method with deep foreground–background modeling. For the clean video, inspired by the foreground–background decomposition, we suggest an unsupervised deep convolutional neural network (CNN) to characterize the foreground and a disentangled spatial–temporal network with an affine operator to exploit the implicit low-rankness of dynamic background. The foreground and background are balanced by a learnable probability mask. Moreover, we learn an optical flow field to enhance temporal consistency. Our deep foreground–background modeling has both high representation abilities brought by the CNN and good data adaptability with interpretable prior knowledge.

For rain streaks, we suggest a learnable total variation (LTV) regularization to faithfully capture the rain direction through the directional smoothness of rain streaks. Instead of tailoring a network to capture the structures of rain streaks [34,35], the hand-crafted LTV is simple yet effective to harness the structures of rain streaks, where the key factor, i.e., the rain direction, is unsupervisedly learned.

We summarize the contributions of this paper as follows:

(1) We propose an unsupervised video rain streaks removal method. The deep foreground–background modeling of the complex clean video and the simple yet effective modeling of structured rain streaks under the basic decomposition framework, which benefit each other in nature, are organically reconciled to enhance the deraining performance. All of the modules have clear physical interpretations and the CNN parameters are unsupervisedly inferred from the observed rainy video.
(2) To address the resulting video deraining model, we develop an efficient ADMM-based algorithm. Vast experiments on both simulated and real-world data validate the superiority of our method over state-of-the-art traditional methods [19–21] and deep learning methods [5,6,28] in terms of qualitative and quantitative results, especially for rain removal and details preserving.

1.1. Related work

Single image restoration, e.g., deraining, has been widely studied in the literature. Traditional methods used prior information to establish optimization models. These techniques included sparse coding [36,37], low-rankness [13], nonlocal means filters [38], convolutional filters [39], Gaussian mixture model [40], fractional-order derivative [41], etc. In recent years, deep learning has made outstanding contributions for the single image deraining. These methods designed neural networks focusing on details preserving [42–44], rain detection [45], rain density estimation [34,46], optimization model unfolding [47], rain generation [48], etc. Meanwhile, many popular deep learning technical tools were applied for image deraining, e.g., transfer learning [49], residual learning [50], continual learning [51], fusion network [52], attention network [53,54], generative adversarial network [55,56], and others [57–60]. Most of these methods learn deep priors from synthetic training datasets, which inevitably neglect the physical structures of rain streaks and clean images.

As compared to single image deraining, video deraining methods effectively utilize temporal information of videos. Garg and Nayar [7,8] used a linear spatial–temporal correlation model to detect rain streaks in videos. Zhang et al. [12] combined temporal and chromatic properties for video deraining. Later methods utilized various models or prior information for video deraining, e.g., sparsity [18], piecewise smoothness [19], low-rankness [13,20], matrix decomposition [14], multiscale convolutional sparse coding [16,21], mixture of Gaussians [15], etc [9,11]. Recently, deep learning methods were proposed for video deraining. These methods mainly utilized CNNs [5,6,27,29] with other elaborately designed techniques including different network structures, e.g., attention module [31], deformable convolution [32], dual-level flow [26], and different learning strategies such as rain generator [28], interaction learning [61], collaborative network [62], self-supervised learning [63], to name but a few [29,30,33,64]. A key module of video deraining is the optical flow, which was widely used in both traditional methods [65] and deep learning methods [63,64]. The optical flow could effectively capture the temporal correlation of videos to boost the deraining performances.

In the literature, there are related unsupervised approaches to obtain video representations. The unsupervised video representation learning [66,67] learned present-past transitions and present-future transitions of videos by feeding the current video context into the network and predict the past and future contexts, in which the learned features can benefit the downstream task, i.e., action recognition. The deep video prior [68–70] was developed to capture the intrinsic video prior in an unsupervised manner. The deep video prior served as prior information that benefits different applications including super-resolution [69], video inpainting [70], dehazing [68], colorization [68], etc. For the task of video deraining, several unsupervised methods have been proposed. Yang et al. [63,71] cleverly utilized the temporal consistency of videos to self-supervisedly learn the clean frame based on its adjacent rainy frames. Zhuang et al. [72] combined the self-supervised deep prior and hand-crafted priors to obtain both good representation and generalization abilities for video deraining. Our method is distinct from these methods because we decompose the complex rainy video into simpler components, i.e., foreground, background, and rain streaks. We can model these basic components more appropriately and easily, and the recovery of each basic component benefits each other interactively to boost the video deraining performance.

2. The proposed method

2.1. Preliminaries

Notations frequently used in this work are listed in Table 1. In addition, the total variation (TV) of a tensor $\mathcal{X} \in \mathbb{R}^{m\times n\times t}$ is defined as $\|\mathcal{X}\|_{\mathrm{TV}} \triangleq \sum_{k=1}^{t}(\sum_{i=1}^{m-1}\sum_{j=1}^{n}|\mathcal{X}^{(k)}(i+1,j)-\mathcal{X}^{(k)}(i,j)|+\sum_{i=1}^{m}\sum_{j=1}^{n-1}|\mathcal{X}^{(k)}(i,j+1)-\mathcal{X}^{(k)}(i,j)|)$. We remark that the symbol k is the counter of the third-mode (the video temporal frames) consistently in this paper. The value of k is taken from the set $\{1,2,\ldots,t\}$, where t denotes the total number of video frames.

We now introduce the degradation model for video deraining. A rainy video is denoted by $\mathcal{O} \in \mathbb{R}^{m\times n\times t}$, where m and n are the spatial sizes and t is the number of frames. We consider the basic rain model as

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

where $\mathcal{C}, \mathcal{R} \in \mathbb{R}^{m\times n\times t}$ denote the clean video and rain streaks, respectively. Real-world videos usually have complex structures, which makes it hard to directly model the holistic video. Motivated by the classical foreground–background decomposition [73,74], we propose to decompose the foreground and background of a video, and then use the deep modeling to respectively model the foreground and background, which allows us to more exactly capture the complex structures of real-world videos.[2] Specifically, the clean video $\mathcal{C}$ can be decomposed as $\mathcal{C} = (\mathbf{1}-\mathcal{P})\circ\mathcal{B}+\mathcal{P}\circ\mathcal{F}$, where $\mathcal{B}$ denotes the background, $\mathcal{F}$ denotes the foreground, and $\mathcal{P}$ is the probability mask. Here, $\circ$ denotes the element-wise product. Therefore, the rainy video is formulated as

$$\mathcal{O} = (\mathbf{1}-\mathcal{P})\circ\mathcal{B}+\mathcal{P}\circ\mathcal{F}+\mathcal{R}, \tag{2}$$

where $\mathbf{0} \le \mathcal{P} \le \mathbf{1}$. Our goal is to estimate the underlying clean video $\mathcal{C} = (\mathbf{1}-\mathcal{P})\circ\mathcal{B}+\mathcal{P}\circ\mathcal{F}$ from the observed $\mathcal{O}$. Next, we respectively introduce our characterizations of the clean video and rain streaks to form the loss function, followed by the ADMM-based algorithm to optimize it.

[2] As far as we know, there is no formal definition of the foreground. However, from an empirical perspective, the foreground in a video can be defined as the objects that are of most importance and activity, or that people pay attention to, e.g., the moving persons or cars in surveillance videos.

Table 1
Notations used in this paper.

Notations	Interpretations
$\mathbf{X} \in \mathbb{R}^{m\times n}$	Matrix
$\mathcal{X} \in \mathbb{R}^{m\times n\times t}$	Third-order tensor
$\mathcal{X}_{ijk}$	The i, j, k-th element of $\mathcal{X}$
$\mathcal{X}^{(i)}$	The ith frontal slice (spatial slice) of $\mathcal{X}$
∇_t	The temporal derivative operator $\nabla_t \mathcal{X} = \mathcal{X}_{(:,:,2:t)} - \mathcal{X}_{(:,:,1:t-1)}$
$\langle \mathcal{A}, \mathcal{B} \rangle$	The tensor inner product $\langle \mathcal{A}, \mathcal{B} \rangle = \sum_{i,j,k} \mathcal{A}(i,j,k)\mathcal{B}(i,j,k)$
$\\|\mathcal{X}\\|_F$	The tensor Frobenius norm $\\|\mathcal{X}\\|_F = \sqrt{\langle \mathcal{X}, \mathcal{X} \rangle} = \sqrt{\sum_{ijk} \mathcal{X}_{ijk}^2}$
$\\|\mathcal{X}\\|_{\ell_1}$	The tensor ℓ_1-norm $\\|\mathcal{X}\\|_{\ell_1} = \sum_{ijk} \|\mathcal{X}_{ijk}\|$
$\mathrm{I}(\mathbf{X}, (x, y))$	The bilinear interpolation, i.e., $\mathrm{I}(\mathbf{X}, (x, y))$ denotes the bilinear interpolation result of $\mathbf{X}$ at the coordinate $(x, y) \in \mathbb{R}^2$
$\texttt{unfold}(\cdot)$	The unfolding operator $\texttt{unfold}(\cdot) : \mathbb{R}^{m\times n\times t} \to \mathbb{R}^{mn\times t}$
$\texttt{fold}(\cdot)$	The folding operator $\texttt{fold}(\cdot) : \mathbb{R}^{mn\times t} \to \mathbb{R}^{m\times n\times t}$
θ_k	The rain direction in the kth frame
$\tau_{\theta_k}(\cdot)$	The affine operator parameterized by rain direction θ_k
$\tau_{\psi}(\cdot)$	The affine operator for background parameterized by $\psi \in \mathbb{R}^{2\times 3\times t}$

2.2. Characterizations of clean video

Due to the moving foreground and dynamic background, the temporal consistency [19,63] of video and the explicit low-rankness of background [16,21,75] rarely exist. To address these issues, we use an unsupervised CNN to capture the foreground and a disentangled spatial–temporal network with an affine operator to depict the dynamic background with implicit low-rankness. The foreground and background are balanced through a learnable probability mask. Moreover, we learn an optical flow to enhance temporal consistency. As compared with the direct modeling of the entire complex video [68–70], our method decomposes the complex video into basic components (i.e., the foreground and background), allowing us to model each component more easily and exactly.

2.2.1. Unsupervised CNN for foreground

The complex moving foreground scenes in videos are hard to be captured by hand-crafted regularizers. Traditional methods assume that the foreground is sparse [76–78] or piecewise smooth [21], thus ℓ_1-norm or TV regularization can be adopted. However, objects like cars often have complex textures, which may not accord with the sparse and smooth assumptions. Only considering ℓ_1-norm or TV without sufficient representation abilities may fail to fully capture the foreground.

In this paper, we propose to model the foreground tensor $\mathcal{F} \in \mathbb{R}^{m\times n\times t}$ using an unsupervised deep CNN [79]. We consider the generative U-Net CNN [80], which can unsupervisedly capture low-level statistics with high representation abilities [68,79]. Specifically, the foreground is obtained through $\mathcal{F} = f_\xi(\mathcal{O})$, where $f_\xi(\cdot)$ is a U-Net CNN [79] parameterized by ξ, and the rainy video $\mathcal{O}$ is the network input. The U-Net has sufficient representation abilities to model the complex scenes of the foreground, which would be difficult to be captured by hand-crafted regularizers. The learnable parameters for obtaining the foreground $\mathcal{F}$ are the U-Net parameters ξ. Hence, the foreground $\mathcal{F}$ is parameterized by ξ and we can naturally use the notation $\mathcal{F}_\xi \triangleq f_\xi(\mathcal{O})$ to denote the foreground parameterized by ξ.

2.2.2. Disentangled spatial–temporal network with affine operator for background

Many traditional methods assume that the background has low-rank structures [16,76–78]. Specifically, given the background tensor $\mathcal{B} \in \mathbb{R}^{m\times n\times t}$, they assume that its unfolding matrix $\mathbf{B} = \texttt{unfold}(\mathcal{B}) \in \mathbb{R}^{mn\times t}$ is low-rank.

However, due to the camera movement, the background $\mathcal{B}$ is not strictly low-rank; see Fig. 2. To exploit the hidden structures of the background, we assume an implicit low-rank structure, i.e., $\mathcal{B} = \tau_\psi(\widetilde{\mathcal{B}})$, where $\widetilde{\mathcal{B}}$ is strictly low-rank and $\tau_\psi(\cdot)$ is the affine operator parameterized by $\psi \in \mathbb{R}^{2\times 3\times t}$, which is related to the camera movement. This is a reasonable and more generalized assumption as compared with the low-rank assumption.

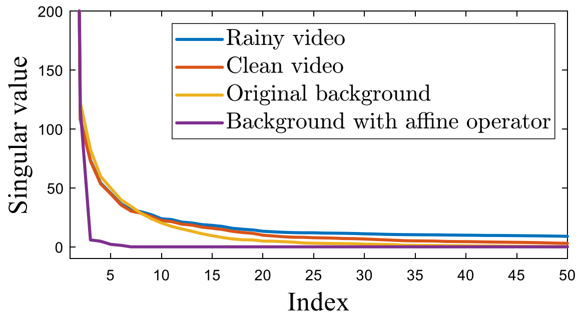

Fig. 2. The singular values of the unfolding matrices of the rainy video $\mathcal{O}$, the clean video $\mathcal{C}$, the original background $\mathcal{B}$, and the background with the affine operator, i.e., $\widetilde{\mathcal{B}}$, on simulated data *Car* in **Case 1**. From the distributions of the singular values, we can observe that the original background $\mathcal{B}$ is implicitly low-rank.

Specifically, we assume that $\text{rank}(\widetilde{\mathbf{B}}) = r$ $(r < \min(mn, t))$, where $\widetilde{\mathbf{B}} = \texttt{unfold}(\widetilde{\mathcal{B}})$ is the mode-3 unfolding matrix of $\widetilde{\mathcal{B}}$. From the low-rank matrix factorization theory, $\widetilde{\mathbf{B}}$ can be factorized as $\widetilde{\mathbf{B}} = \mathbf{SC}$, where $\mathbf{S} \in \mathbb{R}^{mn\times r}$ are coefficients, $\mathbf{C} \in \mathbb{R}^{r\times t}$ are basis, and $r < \min(mn, t)$ is the number of basis. Inspired by the spectral unmixing [81], the coefficients $\mathbf{S}$ and basis $\mathbf{C}$ can be physically interpreted as the spatial abundance and temporal signature, respectively. The matrix factorization motivates us to design a disentangled spatial–temporal network to effectively and efficiently capture the underlying low-rank structure of the background. Specifically, the spatial abundance $\mathbf{S}$ contains spatial information of the video and thus we consider using an untrained U-Net $s_\zeta(\cdot)$ with parameters ζ to capture the complex spatial information. As compared, the temporal signature $\mathbf{C}$ has relatively simpler structures and thus we consider using a simple fully connected network (FCN) $c_\eta(\cdot)$ with parameters η to capture the temporal signature. Consequently, we have $\mathbf{C} = c_\mu(\mathbf{Z})$ and $\mathbf{S} = \texttt{unfold}(s_\zeta(\mathcal{O}))$, where $\mathbf{Z}$ is the random input matrix [82,83] of the FCN and $\mathcal{O}$ is the input of the U-Net. Hence, we have $\widetilde{\mathbf{B}} = \texttt{unfold}(s_\zeta(\mathcal{O}))c_\eta(\mathbf{Z})$. By folding the resulting matrix $\widetilde{\mathbf{B}}$, we can obtain the strictly low-rank tensor $\widetilde{\mathcal{B}} = \texttt{fold}(\widetilde{\mathbf{B}})$.

Next, we introduce the definition of the affine operator. The affine operator $\tau_\psi(\cdot)$ is defined as $(\tau_\psi(\widetilde{\mathcal{B}}))_{ijk} = \mathrm{I}(\widetilde{\mathcal{B}}^{(k)}, (\hat{i}, \hat{j}))$, where

$$\begin{bmatrix} \hat{i} \\ \hat{j} \end{bmatrix} = \begin{bmatrix} \psi_{11k} & \psi_{12k} & \psi_{13k} \\ \psi_{21k} & \psi_{22k} & \psi_{23k} \end{bmatrix} \begin{bmatrix} i \\ j \\ 1 \end{bmatrix} \in \mathbb{R}^2 \tag{3}$$

and $\mathrm{I}(\widetilde{\mathcal{B}}^{(k)}, (\hat{i}, \hat{j}))$ returns the bilinear interpolation result of the matrix $\widetilde{\mathcal{B}}^{(k)}$ at the coordinate $(\hat{i}, \hat{j})$.[3]

Combining the disentangled spatial–temporal network and the affine operator, the background can be finally obtained via

$$\mathcal{B} = \tau_\psi\Big(\texttt{fold}\big(\texttt{unfold}(s_\zeta(\mathcal{O}))c_\eta(\mathbf{Z})\big)\Big). \tag{4}$$

This formulation captures the implicit low-rankness of the background by using the disentangled spatial–temporal network and the affine operator. Here, the learnable parameters are $\omega = \{\psi, \zeta, \eta\}$. For simplicity, we use $\mathcal{B}_\omega \triangleq \tau_\psi\Big(\texttt{fold}\big(\texttt{unfold}(s_\zeta(\mathcal{O}))c_\eta(\mathbf{Z})\big)\Big)$ to denote the background tensor $\mathcal{B}$ parameterized by ω.

2.2.3. Learnable probability mask

We propose to learn a probability mask $\mathcal{P}$ to combine the foreground and background. Specifically, we employ another U-Net $p_\kappa(\cdot)$ to obtain $\mathcal{P}$, i.e., $\mathcal{P} = p_\kappa(\mathcal{O})$, where the rainy video $\mathcal{O}$ is the network input. Since $\mathbf{0} \le \mathcal{P} \le \mathbf{1}$, the Sigmoid function is adopted in the output layer of $p_\kappa(\cdot)$. The probability mask $\mathcal{P}$ combines the foreground and background to obtain the clean video via $\mathcal{C} = (\mathbf{1} - \mathcal{P}) \circ \mathcal{B} + \mathcal{P} \circ \mathcal{F}$. Here, the learnable parameters are the U-Net parameters κ. For simplicity, we use $\mathcal{P}_\kappa \triangleq p_\kappa(\mathcal{O})$ to denote the probability mask tensor parameterized by κ. Based on the basic rain model, the first loss function is the **fidelity loss function**:

$$L_{\text{fidelity}} = \|\mathcal{O} - (\mathbf{1} - \mathcal{P}_\kappa) \circ \mathcal{B}_\omega - \mathcal{P}_\kappa \circ \mathcal{F}_\xi - \mathcal{R}\|_F^2. \tag{5}$$

Next, we form the loss function for the probability mask. First, the probability mask $\mathcal{P}$ should be close to 0 or 1 [84]. Note that $\mathcal{P}$ is not strictly binary as assumed in other models [21], as objects like transparent objects cannot be simply classified as background or foreground. Secondly, we consider a sparse constraint on $\mathcal{P}_\kappa$ to depict the sparsity of the foreground. Thirdly, to distinguish between rain streaks and relatively smoother foreground, we consider the TV regularization on $\mathcal{P}_\kappa$. Therefore, we have the following **loss function for probability mask**:

$$L_{\text{mask}} = \lambda_1 \frac{1}{\|\mathcal{P}_\kappa - 0.5\|_{\ell_1}} + \lambda_2 \|\mathcal{P}_\kappa\|_{\ell_1} + \lambda_3 \|\mathcal{P}_\kappa\|_{\text{TV}}. \tag{6}$$

[3] Here, $\hat{i}$ and $\hat{j}$ may not be integers. Thus, we use the bilinear interpolation to obtain the value of $\widetilde{\mathcal{B}}^{(k)}$ at the coordinate $(\hat{i}, \hat{j})$.

In L_{mask}, the first term encourages the learned probability mask $\mathcal{P}_\kappa$ to be close to 0 or 1.

2.2.4. Learnable optical flow field

The temporal consistency[4] is widely utilized in video deraining [18,19,63]. The temporal consistency can be exploited through minimizing $\|\nabla_t \mathcal{C}\|_{\ell_1}$, where $\mathcal{C}$ is the underlying clean video and ∇_t is the temporal derivative operator [18]. However, due to dynamic objects or camera movements, the temporal consistency is rarely satisfied. Therefore, we introduce a learnable optical flow field to align adjacent frames to enhance the temporal consistency.

More concretely, given a tensor $\mathcal{C} \in \mathbb{R}^{m\times n\times t}$, we propose a learnable optical flow field $\mu \in \mathbb{R}^{p\times 2\times t}$ to align its adjacent frames. Here, the elements of μ represent the spatial shift distance. Each frame of $\mathcal{C}$ is divided into p patches, and the elements in the same patch share the same spatial shifts. Under the optical flow μ, the aligned result $\hat{\mathcal{C}}$ is obtained through $\hat{\mathcal{C}}_{ijk} = \mathrm{I}(\mathcal{C}^{(k)}, (\hat{i}, \hat{j}))$, where $\hat{i} = i + \mu_{v1k}$ and $\hat{j} = j + \mu_{v2k}$. Here, v indicates that the coordinate (i, j) locates in the vth patch in a frame ($v = 1, 2, \ldots, p$). We expect to learn the optical flow field μ such that the $(k+1)$-th frame of $\hat{\mathcal{C}}$ is aligned with the kth frame of $\mathcal{C}$. Therefore, we define the temporal derivative operator with the optical flow tensor μ, denoted by ∇_μ, as $\nabla_\mu \mathcal{C} \triangleq \mathcal{C}_{(:,:,2:t)} - \hat{\mathcal{C}}_{(:,:,1:t-1)}$. We can minimize the ℓ_1-norm of $\nabla_\mu \mathcal{C}$ to depict the temporal consistency. Note that $\mathcal{C} = (\mathbf{1} - \mathcal{P}) \circ \mathcal{B} + \mathcal{P} \circ \mathcal{F}$. Thus, we have the following **temporal consistency loss**:

$$L_{\text{temporal}} = \|\nabla_\mu((\mathbf{1} - \mathcal{P}_\kappa) \circ \mathcal{B}_\omega + \mathcal{P}_\kappa \circ \mathcal{F}_\xi)\|_{\ell_1}. \tag{7}$$

Here, the learnable parameter is the optical flow tensor μ.

2.3. Characterizations of rain streaks

2.3.1. Learnable total variation

Next, we consider the characterizations of rain streaks. Earlier methods assume that rain streaks are vertical, and the directional smoothness can be utilized to remove vertical rain streaks [18]. However, rain streaks often have different directions. Later methods designed techniques to capture different rain directions [17,34]. However, these techniques either need training data [34] or need to manually determine the rain directions [17]. In this paper, we propose the LTV regularization to automatically detect the rain directions.

Specifically, given a rain tensor $\mathcal{R} \in \mathbb{R}^{m\times n\times t}$, the vertical derivative operator ∇_y is defined as $(\nabla_y \mathcal{R})^{(k)} = \mathbf{H} \otimes \mathcal{R}^{(k)}$, where

$$\mathbf{H} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 0 \end{bmatrix} \tag{8}$$

is the vertical derivative convolutional kernel, $\otimes$ is the convolutional operator, and $k = 1, 2, \ldots, t$. Since rain streaks are local smooth along their falling directions (which is known as the directional gradient prior or the directional smoothness [18,19,85]), minimizing $\|\nabla_y \mathcal{R}\|_{\ell_1}$ is helpful for characterizing rain streaks [18]. To capture rain streaks in different directions, we propose t affine operators $\{\tau_{\theta_k}(\cdot)\}_{k=1}^t$ with learnable parameters $\{\theta_k \in \mathbb{R}\}_{k=1}^t$ to modify the kernel $\mathbf{H}$, so that each new kernel $\tau_{\theta_k}(\mathbf{H}) \in \mathbb{R}^{3\times 3}$ can extract the derivative value along the direction θ_k. The new kernel is defined by $(\tau_{\theta_k}(\mathbf{H}))_{ij} = \mathrm{I}(\mathbf{H}, (\hat{i}, \hat{j}))$ $(i, j = 1, 2, 3)$, where

$$\begin{bmatrix} \hat{i} \\ \hat{j} \end{bmatrix} = \begin{bmatrix} \cos(\theta_k) & -\sin(\theta_k) \\ \sin(\theta_k) & \cos(\theta_k) \end{bmatrix} \begin{bmatrix} i \\ j \end{bmatrix}. \tag{9}$$

Equipped with the new kernels $\{\tau_{\theta_k}(\mathbf{H})\}_{k=1}^t$, we define the derivative operator ∇_θ as $(\nabla_\theta \mathcal{R})^{(k)} = (\tau_{\theta_k}(\mathbf{H})) \otimes \mathcal{R}^{(k)}$ ($k = 1, 2, \ldots, t$). Compared with the vertical derivative tensor $\nabla_y \mathcal{R}$, the tensor $\nabla_\theta \mathcal{R}$ considers rain streaks in different directions and therefore is more sparse; see Fig. 4. Thus, by minimizing $\|\nabla_\theta \mathcal{R}\|_{\ell_1}$, the parameter θ_k (which denotes the rain direction in the kth frame) can be unsupervisedly learned, and rain streaks in different directions can be eliminated. The corresponding **LTV loss function** is $L_{\text{LTV}} = \|\nabla_\theta \mathcal{R}\|_{\ell_1}$. The learnable parameter are the rain directions $\theta = \{\theta_k\}_{k=1}^t$.

2.3.2. Sparsity of rain streaks

Rain streaks appear in a few places in the video and thus are sparser than the video content. Therefore, we consider the **sparse loss function** $L_{\text{sparse}} = \|\mathcal{R}\|_{\ell_1}$ to depict such property. Here, the rain tensor $\mathcal{R}$ itself is learnable.

[4] We remark here that the temporal consistency is considered on the clean video $\mathcal{C}$, which is different from the low-rankness considered on the background $\mathcal{B}$ in Section 2.2.2.

2.4. Total loss function

Combining different loss functions, we have the following total loss function by given the observed rainy video $\mathcal{O}$:

$$\begin{aligned}\min_{\substack{\xi,\omega,\kappa\\ \mu,\theta,\mathcal{R}}} & \frac{1}{2}\|\mathcal{O}-(\mathbf{1}-\mathcal{P}_\kappa)\circ\mathcal{B}_\omega-\mathcal{P}_\kappa\circ\mathcal{F}_\xi-\mathcal{R}\|_F^2+\lambda_1/\|\mathcal{P}_\kappa-0.5\|_{\ell_1}+\lambda_2\|\mathcal{P}_\kappa\|_{\ell_1}+\\ & \lambda_3\|\mathcal{P}_\kappa\|_{\mathrm{TV}}+\lambda_4\|\nabla_\mu((\mathbf{1}-\mathcal{P}_\kappa)\circ\mathcal{B}_\omega+\mathcal{P}_\kappa\circ\mathcal{F}_\xi)\|_{\ell_1}+\lambda_5\left\|\nabla_\theta\mathcal{R}\right\|_{\ell_1}+\lambda_6\left\|\mathcal{R}\right\|_{\ell_1}.\end{aligned}\tag{10}$$

In our loss function, the first term is the fidelity term that forces the sum of the recovered clean video and the rain streaks to be close to the rainy video. The term $\lambda_1/\|\mathcal{P}_\kappa-0.5\|_{\ell_1}$ forces the probability mask to be close to 0 or 1. The term $\lambda_2\|\mathcal{P}_\kappa\|_{\ell_1}$ constrains the sparsity of the probability mask since the foreground in videos is sparse in both spatial and temporal dimensions. The term $\lambda_3\|\mathcal{P}_\kappa\|_{\mathrm{TV}}$ constrains the smoothness of the probability mask. The term $\lambda_4\|\nabla_\mu((\mathbf{1}-\mathcal{P}_\kappa)\circ\mathcal{B}_\omega+\mathcal{P}_\kappa\circ\mathcal{F}_\xi)\|_{\ell_1}$ constrains the temporal consistency of the recovered clean video. The term $\lambda_5\left\|\nabla_\theta\mathcal{R}\right\|_{\ell_1}$ constrains the smoothness of rain streaks along the falling direction. The term $\lambda_6\left\|\mathcal{R}\right\|_{\ell_1}$ constrains the sparsity of rain streaks.

Note that the loss function (10) solely uses the rainy video $\mathcal{O}$ without additional training data. However, directly solving the above problem is difficult. We turn to develop an ADMM-based algorithm to update the learnable parameters in a fully unsupervised manner alternately.

2.5. Algorithm

To minimize the loss function (10), we develop an ADMM-based algorithm. We introduce auxiliary variables $\mathcal{V}_1\in\mathbb{R}^{m\times n\times t}$ and $\mathcal{V}_2\in\mathbb{R}^{m\times n\times t}$, and (10) can be rewritten as

$$\begin{aligned}\min_{\substack{\xi,\omega,\kappa,\mu\\ \theta,\mathcal{R},\mathcal{V}_k}} & \frac{1}{2}\|\mathcal{O}-(\mathbf{1}-\mathcal{P}_\kappa)\circ\mathcal{B}_\omega-\mathcal{P}_\kappa\circ\mathcal{F}_\xi-\mathcal{R}\|_F^2+\lambda_1/\|\mathcal{P}_\kappa-0.5\|_{\ell_1}+\lambda_2\|\mathcal{P}_\kappa\|_{\ell_1}+\\ & \lambda_3\|\mathcal{P}_\kappa\|_{\mathrm{TV}}+\lambda_4\|\nabla_\mu((\mathbf{1}-\mathcal{P}_\kappa)\circ\mathcal{B}_\omega+\mathcal{P}_\kappa\circ\mathcal{F}_\xi)\|_{\ell_1}+\lambda_5\left\|\mathcal{V}_1\right\|_{\ell_1}+\lambda_6\left\|\mathcal{V}_2\right\|_{\ell_1},\\ \text{s.t.}\quad & \mathcal{V}_1=\nabla_\theta\mathcal{R},\quad \mathcal{V}_2=\mathcal{R}.\end{aligned}\tag{11}$$

The augmented Lagrangian function corresponding to (11) is

$$\begin{aligned}&\mathcal{L}(\xi,\omega,\kappa,\mu,\theta,\mathcal{R},\mathcal{V}_k,\Lambda_k)=\\ &\frac{1}{2}\|\mathcal{O}-(\mathbf{1}-\mathcal{P}_\kappa)\circ\mathcal{B}_\omega-\mathcal{P}_\kappa\circ\mathcal{F}_\xi-\mathcal{R}\|_F^2+\lambda_1/\|\mathcal{P}_\kappa-0.5\|_{\ell_1}+\lambda_2\|\mathcal{P}_\kappa\|_{\ell_1}+\\ &\lambda_3\|\mathcal{P}_\kappa\|_{\mathrm{TV}}+\lambda_4\|\nabla_\mu(\mathbf{1}-\mathcal{P}_\kappa)\circ\mathcal{B}_\omega+\mathcal{P}_\kappa\circ\mathcal{F}_\xi\|_{\ell_1}+\lambda_5\left\|\mathcal{V}_1\right\|_{\ell_1}+\lambda_6\left\|\mathcal{V}_2\right\|_{\ell_1}+\\ &\frac{\gamma}{2}\left\|\nabla_\theta\mathcal{R}-\mathcal{V}_1\right\|_F^2+\frac{\gamma}{2}\left\|\mathcal{R}-\mathcal{V}_2\right\|_F^2+<\Lambda_1,\nabla_\theta\mathcal{R}-\mathcal{V}_1>+<\Lambda_2,\mathcal{R}-\mathcal{V}_2>.\end{aligned}\tag{12}$$

Here, Λ_1 and Λ_2 are multipliers and γ is the penalty parameter. We can split the joint minimization problem into easier sub-problems under the framework of ADMM.

Update $\mathcal{V}$: The $\mathcal{V}_1$ and $\mathcal{V}_2$ sub-problems are

$$\begin{cases}\min\limits_{\mathcal{V}_1}\frac{\gamma}{2}\left\|\nabla_\theta\mathcal{R}+\Lambda_1/\gamma-\mathcal{V}_1\right\|_F^2+\lambda_5\left\|\mathcal{V}_1\right\|_{\ell_1}\\ \min\limits_{\mathcal{V}_2}\frac{\gamma}{2}\left\|\mathcal{R}+\Lambda_2/\gamma-\mathcal{V}_2\right\|_F^2+\lambda_6\left\|\mathcal{V}_2\right\|_{\ell_1},\end{cases}\tag{13}$$

which can be explicitly solved by $\mathcal{V}_1=Soft_{\frac{\lambda_5}{\gamma}}(\nabla_\theta\mathcal{R}+\Lambda_1/\gamma)$ and $\mathcal{V}_2=Soft_{\frac{\lambda_6}{\gamma}}(\mathcal{R}+\Lambda_2/\gamma)$, where $Soft_a(\cdot)$ denotes the soft-thresholding operator [18].

Update $\{\xi,\omega,\mu,\theta\}$: The variables $\{\xi,\omega,\mu,\theta\}$ can be updated simultaneously in one sub-problem:

$$\begin{aligned}\min_{\xi,\omega,\mu,\theta} & \lambda_4\|\nabla_\mu((\mathbf{1}-\mathcal{P}_\kappa)\circ\mathcal{B}_\omega+\mathcal{P}_\kappa\circ\mathcal{F}_\xi)\|_{\ell_1}+\frac{\gamma}{2}\left\|\nabla_\theta\mathcal{R}-\mathcal{V}_1+(\Lambda_1/\gamma)\right\|_F^2+\\ & \frac{1}{2}\|\mathcal{O}-(\mathbf{1}-\mathcal{P}_\kappa)\circ\mathcal{B}_\omega-\mathcal{P}_\kappa\circ\mathcal{F}_\xi-\mathcal{R}\|_F^2.\end{aligned}\tag{14}$$

Due to the non-convexity of the above sub-problem, we utilize the adaptive moment estimation (Adam) algorithm [86] to solve it. At each iteration of the ADMM-based algorithm, we employ 10 steps of the Adam to update $\{\xi,\omega,\mu,\theta\}$.

Update κ: This κ sub-problem is

$$\begin{aligned}\min_{\kappa} & \frac{1}{2}\|\mathcal{O}-(\mathbf{1}-\mathcal{P}_\kappa)\circ\mathcal{B}_\omega-\mathcal{P}_\kappa\circ\mathcal{F}_\xi-\mathcal{R}\|_F^2+\lambda_1/\|\mathcal{P}_\kappa-0.5\|_{\ell_1}+\\ & \lambda_2\|\mathcal{P}_\kappa\|_{\ell_1}+\lambda_3\|\mathcal{P}_\kappa\|_{\mathrm{TV}}.\end{aligned}\tag{15}$$

Similarly, we employ 10 steps of the Adam to update the U-Net parameters κ in each iteration of the ADMM-based algorithm.

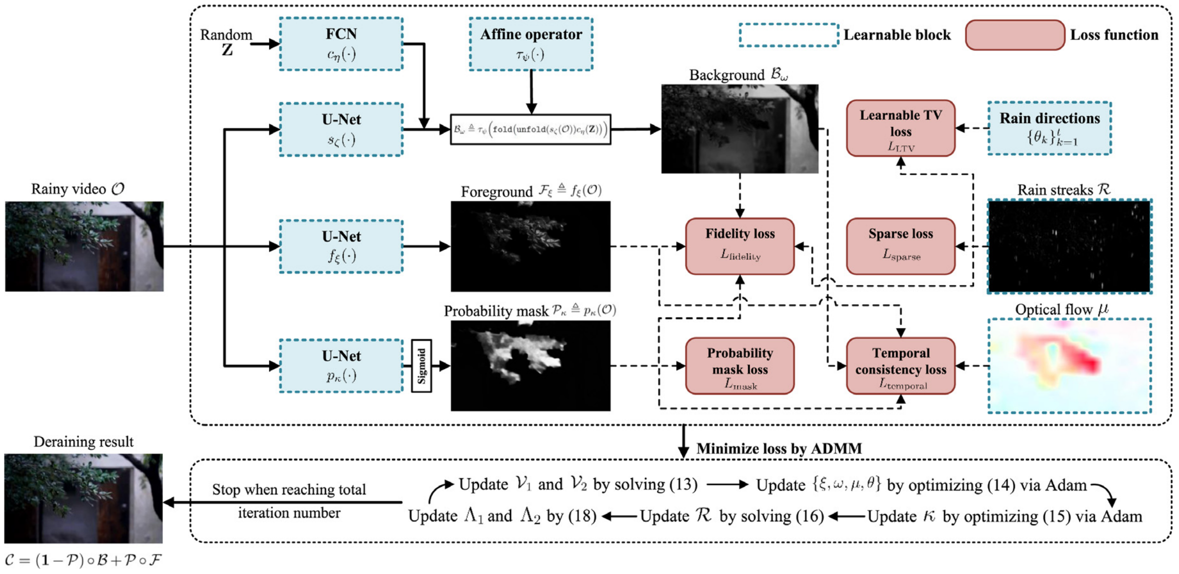
Random Z
FCN $c_\eta(\cdot)$
Affine operator $\tau_\psi(\cdot)$
Learnable block
Loss function
U-Net $s_\zeta(\cdot)$
$\mathcal{B}_\omega \triangleq \tau_\psi(\mathrm{fold}(\mathrm{unfold}(s_\zeta(\mathcal{O}))c_\eta(\mathbf{Z})))$
Background $\mathcal{B}_\omega$
Learnable TV loss L_{LTV}
Rain directions $\{\theta_k\}_{k=1}^t$
Rainy video $\mathcal{O}$
Foreground $\mathcal{F}_\xi \triangleq f_\xi(\mathcal{O})$
U-Net $f_\xi(\cdot)$
Fidelity loss L_{fidelity}
Sparse loss L_{sparse}
Rain streaks $\mathcal{R}$
Probability mask $\mathcal{P}_\kappa \triangleq p_\kappa(\mathcal{O})$
U-Net $p_\kappa(\cdot)$
Sigmoid
Probability mask loss L_{mask}
Temporal consistency loss L_{temporal}
Optical flow μ
Minimize loss by ADMM
Deraining result
Stop when reaching total iteration number
Update $\mathcal{V}_1$ and $\mathcal{V}_2$ by solving (13)
Update $\{\xi,\omega,\mu,\theta\}$ by optimizing (14) via Adam
Update Λ_1 and Λ_2 by (18)
Update $\mathcal{R}$ by solving (16)
Update κ by optimizing (15) via Adam
$\mathcal{C} = (\mathbf{1}-\mathcal{P})\circ\mathcal{B}+\mathcal{P}\circ\mathcal{F}$

Fig. 3. The overall flowchart of our video deraining method. We use a U-Net to capture the foreground $\mathcal{F}_\xi$ and a disentangled spatial–temporal network with the affine operator to capture the background $\mathcal{B}_\omega$. Meanwhile, we learn a probability mask $\mathcal{P}_\kappa$ to combine the foreground and background. Moreover, we learn the optical flow field μ to enhance temporal consistency. The rain directions $\{\theta_k\}_{k=1}^t$ are learned by minimizing the LTV loss. The resulting loss function (10), which only utilizes the observed rainy video, is minimized by using the ADMM-based algorithm. Here, the structure of the U-Net CNN directly follows [79] and hence is omitted.

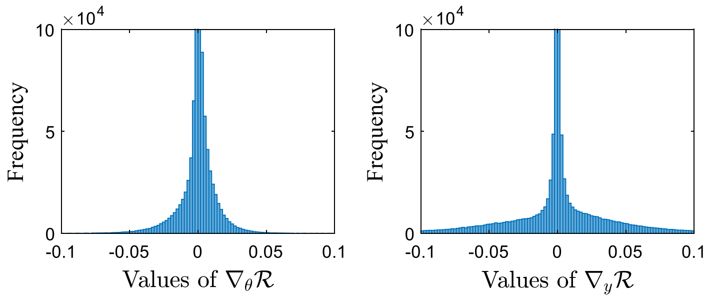

Fig. 4. The frequency of derivative values of real-world rain streaks captured by the camera in black background [16]. (a) The elements of $\nabla_\theta\mathcal{R}$, where the rain directions $\{\theta_k\}_{k=1}^t$ are learned by using the proposed LTV. (b) The elements of $\nabla_y\mathcal{R}$. $\nabla_\theta\mathcal{R}$ is more sparse than $\nabla_y\mathcal{R}$, which reveals the effectiveness of LTV regularization.

Update $\mathcal{R}$: The $\mathcal{R}$ sub-problem can be split into t sub-problems of the frontal slices:

$$\min_{\mathcal{R}^{(k)}} \frac{\gamma}{2}\left\|\tau_{\theta_k}(\mathbf{H})\otimes\mathcal{R}^{(k)} - (\mathcal{V}_1-(\Lambda_1/\gamma))^{(k)}\right\|_F^2 + \frac{\gamma}{2}\left\|\mathcal{R}^{(k)}-(\mathcal{V}_2-(\Lambda_2/\gamma))^{(k)}\right\|_F^2 + \frac{1}{2}\|\mathcal{O}^{(k)} - ((\mathbf{1}-\mathcal{P}_\kappa)\circ\mathcal{B}_\omega + \mathcal{P}_\kappa\circ\mathcal{F}_\xi)^{(k)} - \mathcal{R}^{(k)}\|_F^2, \quad (k=1,2,\ldots,t), \tag{16}$$

which can be explicitly solved by

$$\mathcal{R}^{(k)} = \mathcal{F}^{-1}\Big((\mathcal{F}(\mathcal{O}^{(k)} - ((\mathbf{1}-\mathcal{P}_\kappa)\circ\mathcal{B}_\omega + \mathcal{P}_\kappa\circ\mathcal{F}_\xi)^{(k)} + \gamma(\mathcal{V}_2-(\Lambda_2/\gamma))^{(k)}) + \gamma\overline{\mathcal{F}(\tau_{\theta_k}(\mathbf{H}))}\mathcal{F}((\mathcal{V}_1-(\Lambda_1/\gamma))^{(k)}))(1+\gamma+\gamma\overline{\mathcal{F}(\tau_{\theta_k}(\mathbf{H}))}\mathcal{F}(\tau_{\theta_k}(\mathbf{H})))^{-1}\Big), \tag{17}$$

where $\mathcal{F}$ is Fourier transform operator and $\mathcal{F}^{-1}$ is the inverse operator of $\mathcal{F}$. $\overline{\mathbf{X}}$ is the conjugate matrix of $\mathbf{X}$.

Update Λ: Λ_1 and Λ_2 are updated by

$$\Lambda_1 = \Lambda_1 + \gamma(\nabla_\theta\mathcal{R} - \mathcal{V}_1),\ \ \Lambda_2 = \Lambda_2 + \gamma(\mathcal{R} - \mathcal{V}_2). \tag{18}$$

The overall flowchart of our deraining method is illustrated in Fig. 3. We remark that the main difference between the proposed algorithm and the classical ADMM algorithm is that our algorithm includes the update of the neural network parameters in the $\{\xi, \omega, \mu, \theta\}$ sub-problem and the κ sub-problem. Due to the high nonlinearity and non-convexity of the above sub-problems, we consider using the Adam algorithm to solve these sub-problems, which is easy to implement under the modern deep learning frameworks.

3. Experiments

3.1. Settings

3.1.1. Datasets and evaluation metrics

We first introduce the datasets used in experiments. Since our method is an unsupervised method by solely using the rainy video, no training datasets are needed. We next introduce the testing datasets. We consider five testing datasets. Four of them are synthetic datasets (denoted as **Cases 1–4**) and the other one is a real-world dataset:

- **Case 1** contains 10 pairs of rainy videos and clean videos. Among the 10 clean videos, three of them are publicly available[5,6] and the rests are captured by a digital camera, containing diverse foreground scenes and dynamic background. The sizes of these clean videos are $240 \times 320 \times 50$. We generate synthetic rain streaks on these clean videos using motion blurring. The rainy videos in **Case 1** involve light rain streaks and the rain directions are sampled from $[-30^\circ, 30^\circ]$, where rain directions are different in different frames but are the same in a single frame.
- **Case 2** contains 10 pairs of rainy videos and clean videos, where the clean videos are the same as **Case 1** and rainy videos involve heavy rain streaks and the rain directions are sampled from $[-40^\circ, 40^\circ]$. The rain directions are different in different frames and are also different in a single frame.
- **Case 3** contains 10 pairs of rainy videos and clean videos, where the clean videos are the same as **Case 1**. The rainy videos contain synthetic rain streaks generated by photorealistic render techniques in [87] and rain streaks are heavy with various directions in a single frame.
- **Case 4** contains 4 pairs of rainy videos and clean videos in the benchmark NTURain (a) dataset [5]. The clean videos are captured by a panning unstable camera and rain streaks are generated with the editing software Adobe After Effects [5].
- For real-world data, we collect six publicly available real-world rainy videos[7] with various types of rain streaks and video scenes to test the effectiveness of our method in real rain scenarios.

We use the peak signal-to-noise ratio (PSNR) and structure similarity (SSIM) as the evaluation metrics. Higher values of PSNR and SSIM indicate better performances. We convert the videos from the RGB color space to the YCbCr[8] color space and then perform deraining on the Y channel.

3.1.2. Compared methods

We compare our method with different types of video deraining methods, including traditional model-based video deraining methods (Kim et al. [20], FastDeRain [19], and OTMS-CSC [21]) and deep learning-based video deraining methods (SpacCNN [5], J4RNet [6], and S2VD [28]). Moreover, we include two representative image deraining methods (PReNet [57] and SSIR [88]) into comparisons. SpacCNN, J4RNet, and PReNet are supervised methods. S2VD and SSIR are semi-supervised methods.

The parameters of compared traditional methods (Kim et al. [20], FastDeRain [19], and OTMS-CSC [21]) are adaptively tuned for different rainy videos to obtain the best PSNR value. For deep learning methods (SpacCNN [5], J4RNet [6], S2VD [28], PReNet [57], and SSIR [88]), we use the pre-trained model provided by the authors. We hope to note that the fundamental difference between these deep learning methods and our method is (semi-) supervised learning v.s. unsupervised learning. The purpose of the comparison with these (semi-) supervised methods is to highlight the better data adaptability of our method for diverse rain scenarios. From this perspective, using re-training would defeat this purpose, and is impractical in real scenarios without paired training data. Thus, we directly use the pre-trained models of these (semi-) supervised methods. We remark that the pre-trained models of SpacCNN and S2VD are trained on the NTURain dataset [5], while the pre-trained model of J4RNet is trained on the RainSynLight25 and RainSynComplex25 datasets [6]. The pre-trained models of PReNet and SSIR are trained on the R100L dataset [45] and Rain1400 dataset [43], respectively.

3.1.3. Implementation details

In this subsection, we introduce the implementation details of our method from the following perspectives:

- **Model hyperparameters setting** The model hyperparameters of our method are $\{\lambda_i\}_{i=1}^{6}$. In all experiments, we fix $\lambda_1 = 500$, $\lambda_3 = 0.002$, and $\lambda_5 = 0.02$ and select λ_2, λ_4, and λ_6 from the candidate sets $\{2 \times 10^{-4}, 4 \times 10^{-4}, 6 \times 10^{-4}\}$, $\{0.5 \times 10^{-2}, 1 \times 10^{-2}, 1.5 \times 10^{-2}\}$, and $\{2 \times 10^{-3}, 4 \times 10^{-3}, 6 \times 10^{-3}\}$ respectively to obtain the best PSNR value (simulated data) or visual result (real-world data).

5 http://trace.eas.asu.edu/yuv/
6 http://jacarini.dinf.usherbrooke.ca/dataset2014/
7 https://github.com/hotndy/SPAC-SupplementaryMaterials
8 https://en.wikipedia.org/wiki/YCbCr

Table 2
The average quantitative results by different methods on simulated rainy videos in different datasets. PReNet and SSIR are image deraining methods while other methods are video deraining methods. The **best** and <u>second-best</u> values are highlighted.

Dataset	Metric	Rainy	PReNet	SSIR	Kim et al.	FastDeRain	OTMS-CSC	SpacCNN	J4RNet	S2VD	Ours
Case 1	PSNR	27.34	30.77	24.64	30.36	33.22	<u>34.92</u>	34.75	28.83	28.51	**41.23**
	SSIM	0.8520	0.9328	0.8723	0.9020	0.9290	<u>0.9604</u>	0.9554	0.9124	0.8796	**0.9894**
Case 2	PSNR	21.05	27.35	24.19	26.78	29.40	29.76	<u>29.88</u>	24.85	21.95	**39.39**
	SSIM	0.6409	0.8461	0.8069	0.8056	0.8676	<u>0.8997</u>	0.8859	0.7976	0.6812	**0.9846**
Case 3	PSNR	21.83	25.18	25.27	26.72	<u>30.21</u>	28.47	28.43	28.05	24.20	**38.39**
	SSIM	0.7929	0.8575	0.8800	0.8542	0.8872	0.9152	0.8895	<u>0.9083</u>	0.8570	**0.9823**
Case 4	PSNR	28.74	30.47	22.33	31.32	31.01	31.61	31.95	28.86	**36.18**	<u>34.05</u>
	SSIM	0.9355	0.9389	0.8640	0.9339	0.9272	0.9523	0.9389	0.9259	**0.9630**	<u>0.9556</u>

- **Algorithm hyperparameters setting** In all experiments, we set the total iteration number of the ADMM-based algorithm to 5000. The penalty parameter γ is set to 1. The iteration number, learning rate, and weight decay of the Adam algorithm for solving sub-problems (14) and (15) are set to 10, 0.001, and 0.0001, respectively.
- **Network structures** For the modeling of the temporal signature, we use the classical FCN. The number of layers of the FCN is 4. The nonlinear activation function of the FCN is the ReLU. For the modeling of the foreground, the probability mask, and spatial abundance of the background, we use classical U-Net structures [79,80] with skip connections. The number of downsampling layers and upsampling layers is 3. Please see more details for the structures of the classical U-Net in [79,80]. Moreover, we use the default normal distribution in PyTorch[9] to initialize the network parameters.
- **Platforms** The proposed method is implemented with PyTorch 1.7.0 on an Intel i9 CPU and RTX 3060 GPU. The compared methods Kim et al. FastDeRain, and OTMS-CSC are implemented with MATLAB 2019b. SpacCNN, J4RNet, S2VD, PReNet, and SSIR are implemented with PyTorch 1.7.0.

Next, we discuss the influence of the model hyperparameters on the deraining performance. In the proposed deraining model, the hyperparameters, which balance different loss terms, are $\{\lambda_i\}_{i=1}^6$. To comprehensively investigate the influences of different hyperparameters, we conduct experiments on the synthetic data *Car* and *Building* in **Case 1** with different values of hyperparameters; see Fig. 10. Here, we investigate the influence of each hyperparameter by changing its value and fixing the others. We can observe that it is important to set a suitable value of hyperparameters to obtain good performances, and our method is relatively more robust to the values of λ_1, λ_3, and λ_5. Motivated by these observations, in the main experiments, we fix $\lambda_1 = 500$, $\lambda_3 = 0.002$, and $\lambda_5 = 0.02$ and select λ_2, λ_4, and λ_6 from the candidate sets $\{2 \times 10^{-4}, 4 \times 10^{-4}, 6 \times 10^{-4}\}$, $\{0.5 \times 10^{-2}, 1 \times 10^{-2}, 1.5 \times 10^{-2}\}$, and $\{2 \times 10^{-3}, 4 \times 10^{-3}, 6 \times 10^{-3}\}$ respectively to obtain the best PSNR value.

3.2. Experimental results

The quantitative results for simulated data are shown in Tables 2. For **Cases 1–3**, our method considerably outperforms compared methods in terms of PSNR and SSIM. The distribution shift between testing data and training data makes supervised deep learning-based video deraining methods [5,6,28] hard to handle the complex scenarios in **Cases 1–3**, resulting in less PSNR and SSIM values. Our unsupervised approach only uses the rainy video and has better data adaptability for various datasets with different rain scenarios. Meanwhile, image deraining methods PReNet and SSIR cannot utilize temporal information of videos. Our video deraining method considers the temporal consistency, and thus our method is obviously superior to single image deraining methods for video deraining.

The performance of S2VD is slightly better than our method on **Case 4** (i.e., the NTURain testing dataset [5]). The possible reason is that the S2VD was pre-trained using the NTURain training dataset [5], and the NTURain training dataset and the NTURain testing dataset may follow the same distribution. For wild datasets that do not follow the same distribution as the NTURain training dataset (e.g., **Cases 1–3**), the performance of S2VD degrades. Our method outperforms S2VD in **Cases 1–31**, which validates the better data adaptability of our method. Compared with (semi-) supervised learning methods, our unsupervised method learns different CNN parameters for different rainy videos based on the physical interpretations of foreground, background, and rain streaks. Thus, our method can more flexibly adapt to diverse rain scenarios.

The visual results on simulated data are illustrated in Figs. 5–8. We can observe that our method well removes the complex rain streaks and preserve the fine details in the videos. This is due to the comprehensive considerations of the structures of the foreground, background, and rain streaks, where both the good data adaptability of hand-crafted priors and the representation abilities of CNNs are well utilized.

The visual results on real-world data are illustrated in Fig. 9. We can observe that our unsupervised method shows good data adaptability for different real-world rain scenarios as compared with (semi-) supervised methods. Specifically,

9 https://pytorch.org/docs/stable/nn.init.html

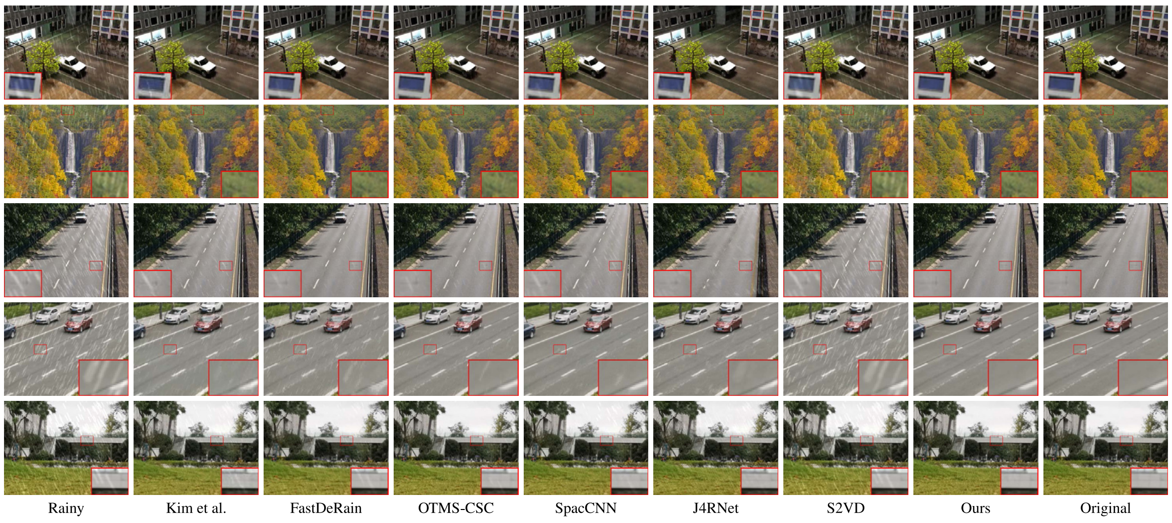

Fig. 5. The deraining results by different methods on simulated data *Truck*, *Waterfall*, *Highway*, *Car*, and *Building* in **Case 1**. (The images are best viewed in full-screen mode.).

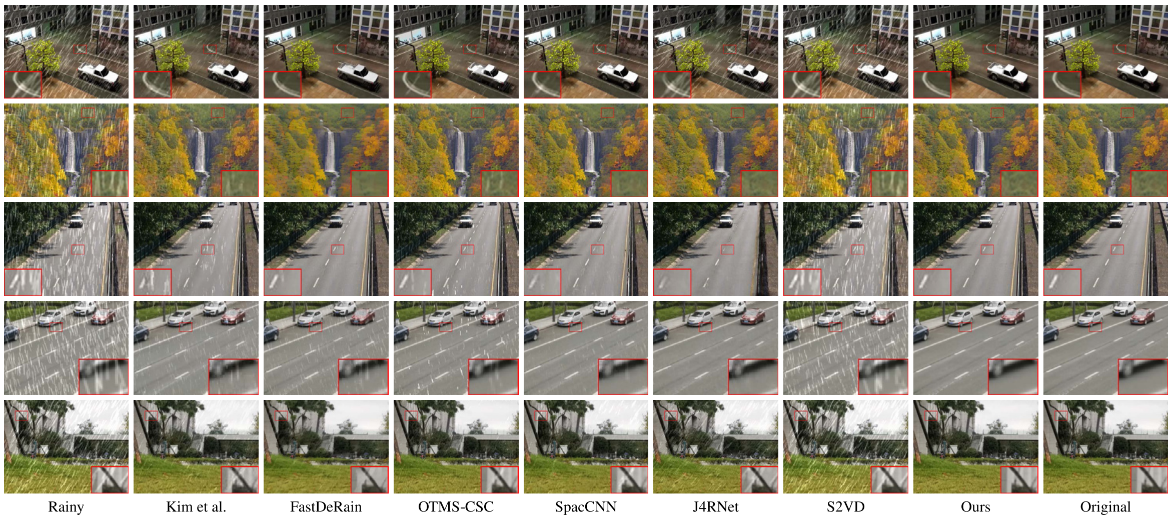

Fig. 6. The deraining results by different methods on simulated data *Truck*, *Waterfall*, *Highway*, *Car*, and *Building* in **Case 2**. (The images are best viewed in full-screen mode.).

our method can well remove the complex rain streaks in the videos and also preserve the edges and details. Meanwhile, our method has stable performances for both heavy rain (the images in the 2nd and 4th rows in Fig. 9) and light rain (other images in Fig. 9). As compared, the (semi-) supervised methods, which depend on training data, unavoidably suffer from domain shift between real-world data and training data, limiting their performances in real scenarios. Our unsupervised method does not depend on training data, and thus it can more flexibly handle diverse real-world complex rain scenarios.

Next, we discuss the effectiveness of our video deraining method for subsequent visual tasks, e.g., object detection and instance segmentation. We first use different video deraining methods to process the rainy video *Truck* in **Case 1** for deraining. Then, we selected three video frames in the deraining results to conduct the object detection and instance segmentation experiments. By following classical object detection and instance segmentation methods [89], which use the whole image as the network input, we separately feed the selected frames (images) into the mask R-CNN [89] pre-trained in the PyTorch library to obtain the object detection and instance segmentation results, as shown in Fig. 14. We can observe that rain streaks indeed seriously affect the accuracy of detection and segmentation according to the results on the rainy video. Meanwhile, the detection and segmentation results with the proposed deraining method are the best among all results, which reveals the superiority of our method over compared video deraining methods for subsequent visual tasks.

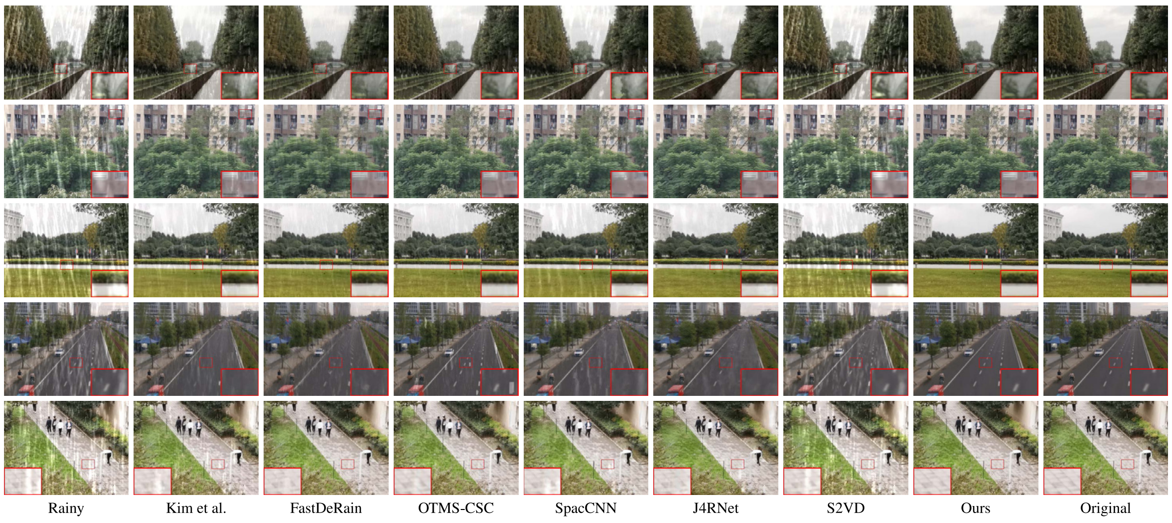

Fig. 7. The deraining results by different methods on simulated data *Bridge*, *Window*, *Square*, *Road*, and *Pedestrian* in **Case 3**. (The images are best viewed in full-screen mode.).

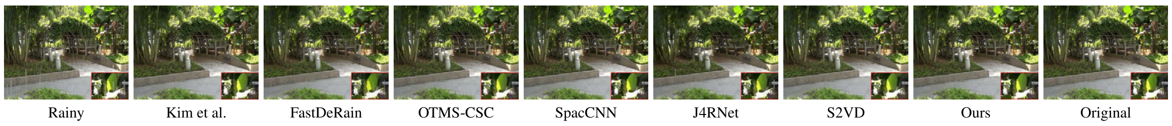

Fig. 8. The deraining results by different methods on a rainy video in **Case 4**. (The images are best viewed in full-screen mode.).

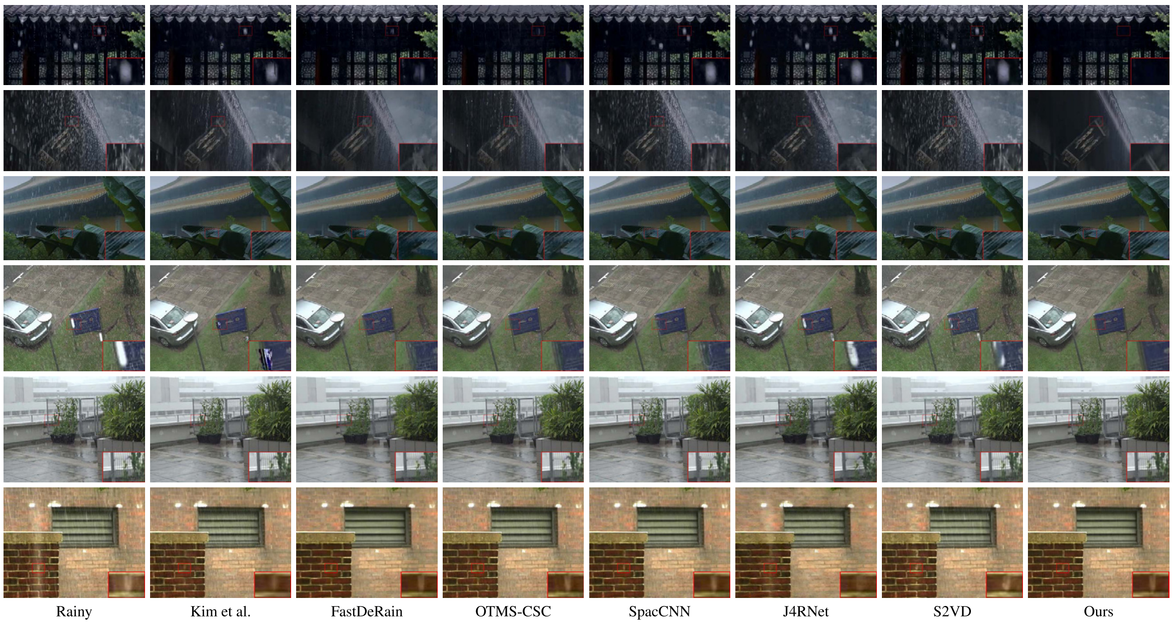

Fig. 9. The deraining results on real-world data. The size of these videos from top to down is 400 × 706 × 20, 400 × 700 × 20, 400 × 676 × 20, 480 × 640 × 20, 480 × 640 × 20, and 288 × 368 × 20. (The images are best viewed in full-screen mode.).

Since our method adaptively learns different CNN parameters for each observed rainy video in an unsupervised manner, it is relatively time-consuming. We report the running time of different methods in Table 5. Future work on accelerating the algorithm, e.g., by using the supervised pre-training [90] or meta learning [91], would be helpful to alleviate the time-consuming limitation.

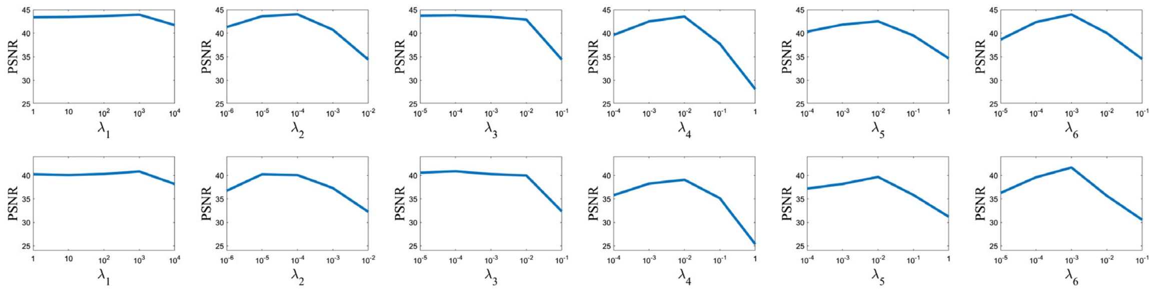

Fig. 10. The PSNR values with respect to the values of different hyperparameters on simulated data *Car* (top row) and *Building* (bottom row) in **Case 1**.

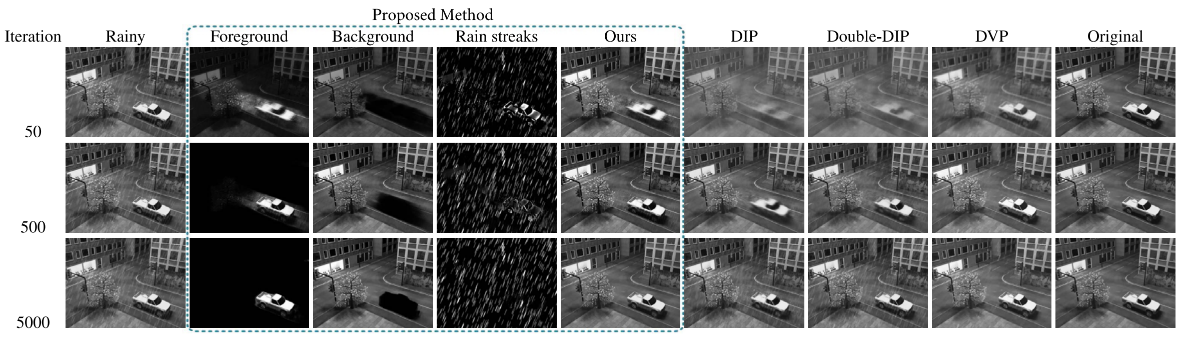

Fig. 11. Illustrations of the foreground, the background, the rain streaks, and the deraining result at different iteration steps by using the proposed method on simulated data *Truck* in **Case 1** and the deraining results by DIP [79], Double-DIP [84], and DVP [68]. Our method decomposes the rainy video into the foreground, background, and rain streaks to more accurately model each component.

3.3. Relations with deep image prior

Compared with deep image prior (DIP) family methods [68,79,84], we decompose the rainy video into the foreground, background, and rain streaks to better model each component. In Fig. 11, we show the foreground, the background, and rain streaks obtained by our method at different iteration steps. Meanwhile, we use the DIP [79], the deep video prior (DVP)[10] [68], and the Double-DIP [84] as baselines. Our method accurately separates different components, while DIP, DVP, and Double-DIP fail to separate the rain streaks. Although DIP can correct flickering artifacts in videos [68], rain streaks have regular structures which cannot be eliminated by the DIP. Our method comprehensively considers foreground modeling, background modeling, and rain streaks modeling to achieve better performance.

Meanwhile, although DIP has been applied for related low-level visual tasks, i.e., the foreground–background image separation by using the Double-DIP [84], we remark that the Double-DIP [84] fails to separate the clean video and rain streaks; see Fig. 11. For video deraining, we elaborately design the modeling of background, foreground, and rain streaks, where the modeling of basic components, e.g., structured rain streaks and the low-rank background factorization are significantly different from Double-DIP [84]. Thus, the advantages of our modeling over Double-DIP [84] for video deraining are clear and significant.

3.4. Comparisons with other unsupervised methods

In this section, we compare our method with other unsupervised/self-supervised video deraining methods (SLDNet [63] and Zhuang et al. [72]) on the NTURain real-world dataset [5]. We report both visual results and quantitative results in Fig. 15. Since there is no ground-truth for real-world data, we consider the non-reference metrics Naturalness Image Quality Evaluator (NIQE) and Perception-based Image Quality Evaluator (PIQE) [25] to measure the quality of the deraining results (Lower NIQE and PIQE values indicate better performances). Since we are unable to successfully run the code of SLDNet, the reported results of SLDNet are provided by the authors from email communications. From Fig. 15, we can observe that our method obtains the best quantitative results. Visually, our method preserves the video details better than other methods. The superior performance of our method is attributed to the elaborate modeling of foreground, background, and rain streaks, which lead to more promising performances on real-world rain scenarios.

[10] We use the GCANet [58] as the single image deraining method, which is needed by the DVP framework.

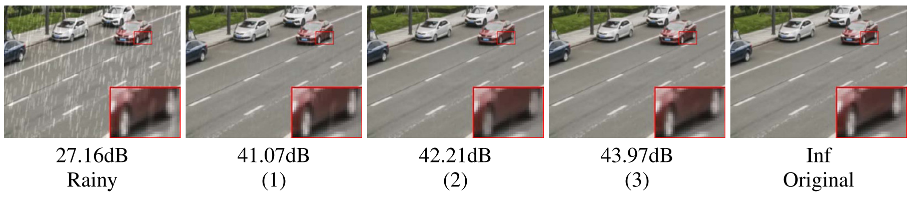

Fig. 12. The deraining results by our method with different foreground priors. (1) Without foreground priors. (2) With foreground TV prior. (3) Our method.

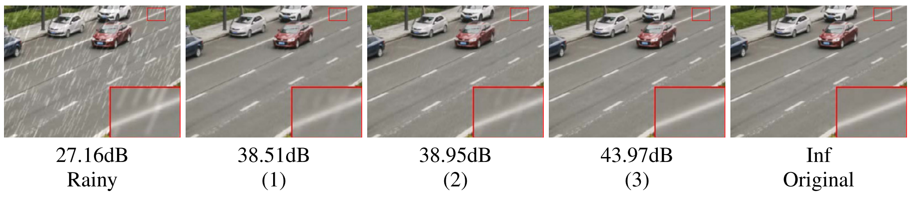

Fig. 13. The deraining results by our method with different background modeling methods. (1) Without affine operator. (2) Without low-rank prior. (3) Our method.

Table 3
The quantitative results by our method with different rain streaks modeling methods on simulated data *Car* in **Case 1**.

Metric	(1)	(2)	(3)	Ours
PSNR	38.24	40.19	33.89	**43.97**
SSIM	0.9803	0.9871	0.9835	**0.9933**

(1) Without LTV loss $\|\nabla_\theta \mathcal{R}\|_{\ell_1}$. (2) Without LTV loss $\|\nabla_\theta \mathcal{R}\|_{\ell_1}$, with $\|\nabla_y \mathcal{R}\|_{\ell_1}$. (3) Without $\|\mathcal{R}\|_{\ell_1}$.

3.5. *Ablation study*

3.5.1. *Contribution of foreground modeling*

We employ a U-Net $f_\xi(\cdot)$ to unsupervisedly capture the foreground $\mathcal{F}$. To test its effectiveness, we consider two baselines. The first one is our method without the U-Net $f_\xi(\cdot)$, i.e., without foreground priors. The second one is our method without the U-Net $f_\xi(\cdot)$ but with an additional foreground smooth prior (i.e., with the loss function $\|\mathcal{F}\|_{\text{TV}}$) [21]. The results on simulated data *Car* in **Case 1** are shown in Fig. 12. The U-Net can more faithfully preserve the details of the foreground compared with hand-crafted TV regularization. This reveals that the deep U-Net brings good representation abilities to better characterize the complex foreground scenes.

3.5.2. *Contribution of background modeling*

We employ a disentangled spatial–temporal network with the affine operator to capture the dynamic background. To test the effectiveness, we consider two baselines. The first one is our method without the affine operator $\tau_\psi(\cdot)$. The second one is our method without the low-rank prior, i.e., we directly use a U-Net to model $\mathcal{B}$ without low-rankness and the affine operator. The results on *Car* in **Case 1** are shown in Fig. 13. We can observe that the affine operator and the low-rank prior are both important in our deraining framework.

3.5.3. *Contribution of rain streaks modeling*

We consider the directional gradient prior and the sparse prior for rain streaks modeling. To test the effectiveness, we consider three baselines. The first one is our method without the LTV loss $\|\nabla_\theta \mathcal{R}\|_{\ell_1}$. The second one is our method without the LTV loss but with the vertical gradient prior, i.e., with the loss $\|\nabla_y \mathcal{R}\|_{\ell_1}$. The third one is our method without the sparse loss $\|\mathcal{R}\|_{\ell_1}$. The results are shown in Table 3. We can observe that the LTV loss and the sparse loss are both important in our method.

3.5.4. *Contribution of probability mask loss*

Next we analyze the influence of the probability mask loss L_{mask}, which includes the close-to-binary constraint $1/\|\mathcal{P}_\kappa - 0.5\|_{\ell_1}$, the sparse constraint $\|\mathcal{P}_\kappa\|_{\ell_1}$, and the smooth constraint $\|\mathcal{P}_\kappa\|_{\text{TV}}$. We remove each of these terms and show the quantitative results in Table 4. We can observe that each term in L_{mask} contributes to the good performance of our method.

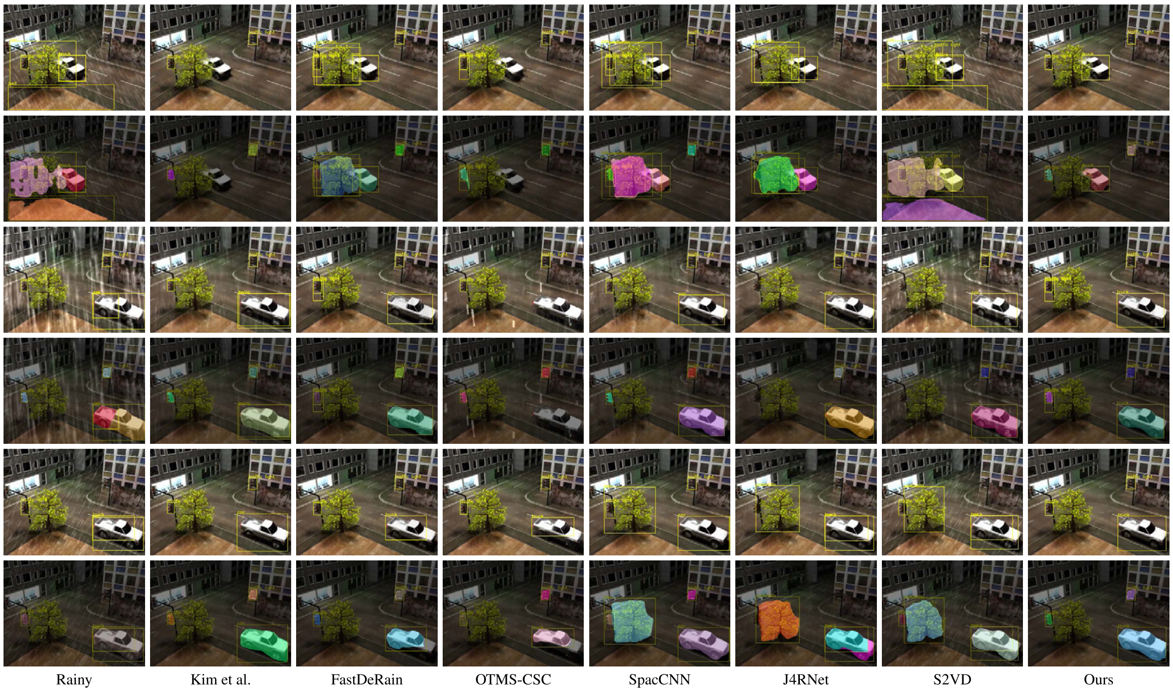

Fig. 14. Three frames of the deraining results (the 1st, 3rd, and 5th rows) by different methods on *Truck* in **Case 1** and the corresponding object detection and instance segmentation results (the 2nd, 4th, and 6th rows) using the mask R-CNN. .

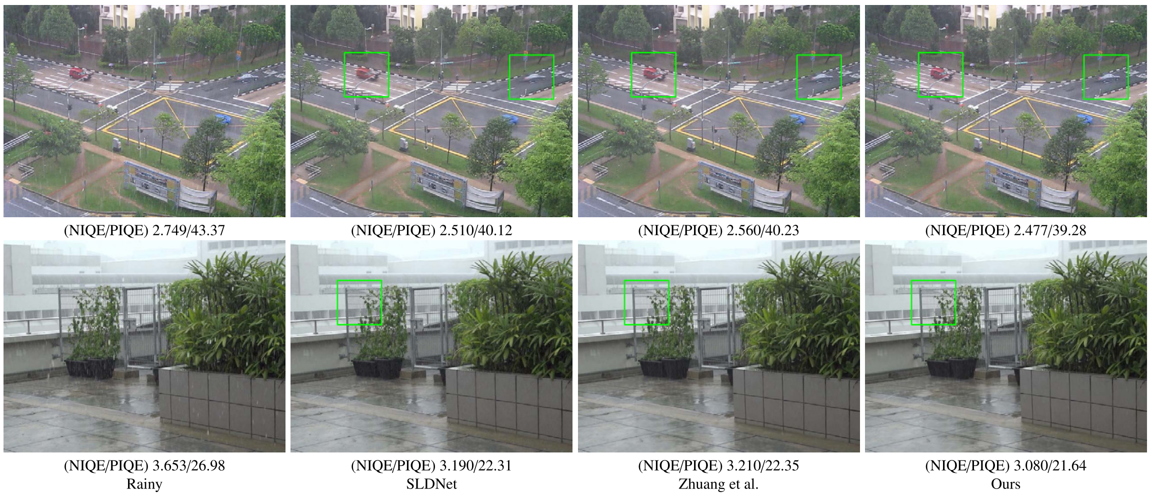

Fig. 15. The deraining results by different methods on two real-world rainy videos in the NTURain dataset.

Table 4
The influence of terms related to the probability mask loss L_{mask} on simulated data *Car* in **Case 1**.

Metric	(1)	(2)	(3)	Ours
PSNR	42.02	40.26	41.15	**43.97**
SSIM	0.9899	0.9853	0.9872	**0.9933**

(1) Without $1/\|\mathcal{P}_\kappa - 0.5\|_{\ell_1}$. (2) Without $\|\mathcal{P}_\kappa\|_{\ell_1}$. (3) Without $\|\mathcal{P}_\kappa\|_{\text{TV}}$.

Table 5
The average running time (s) of different methods in different cases.

Dataset	PReNet	SSIR	Kim et al.	FastDeRain	OTMS-CSC	SpacCNN	J4RNet	S2VD	Ours
Case 1	7.65	12.57	1153.21	4.23	120.32	123.35	230.11	2.03	1096.32
Case 2	7.75	12.65	1123.01	4.01	121.65	127.99	245.19	2.10	1089.07
Case 3	7.71	12.42	1151.96	4.29	125.13	121.59	238.10	2.22	1081.88
Case 4	12.36	20.11	1849.36	7.21	207.02	201.35	366.59	3.91	1502.13

4. Conclusion

This paper proposes an unsupervised video deraining method by solely using the observed rainy video. The deep foreground–background modeling more appropriately exploits the complex structures of the clean video with compact representation abilities and the simple yet effective LTV regularization elegantly captures the structures of rain streaks. The modeling of the clean video and rain streaks organically integrate to enhance the deraining performance. As a result, our method can effectively remove the rain streaks in videos and preserve the image details, as validated by extensive experiments on both synthetic data and real-world data.

Data availability

Data will be made available on request.

Acknowledgments

This research is supported by NSFC, China (No. 12171072, 62131005) and the National Key Research and Development Program of China (No. 2020YFA0714001).

References

[1] C. Feichtenhofer, X3D: Expanding architectures for efficient video recognition, in: IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR, 2020, pp. 200–210.
[2] S. Beery, G. Wu, V. Rathod, R. Votel, J. Huang, Context R-CNN: Long term temporal context for per-camera object detection, in: IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR, 2020, pp. 13072–13082.
[3] Y. Zhang, Z. Qiu, T. Yao, C.-W. Ngo, D. Liu, T. Mei, Transferring and regularizing prediction for semantic segmentation, in: IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR, 2020, pp. 9618–9627.
[4] C.H. Bahnsen, T.B. Moeslund, Rain removal in traffic surveillance: Does it matter? IEEE Trans. Intell. Transp. Syst. 20 (8) (2019) 2802–2819.
[5] J. Chen, C.-H. Tan, J. Hou, L.-P. Chau, H. Li, Robust video content alignment and compensation for rain removal in a CNN framework, in: IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR, 2018, pp. 6286–6295.
[6] J. Liu, W. Yang, S. Yang, Z. Guo, Erase or fill? Deep joint recurrent rain removal and reconstruction in videos, in: IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR, 2018, pp. 3233–3242.
[7] K. Garg, S. Nayar, Detection and removal of rain from videos, in: IEEE/CVF Conference on Computer Vision and Pattern Recognition, Vol. 1, CVPR, 2004.
[8] K. Garg, S. Nayar, When does a camera see rain? in: IEEE International Conference on Computer Vision, Vol. 2, ICCV, 2, 2005, pp. 1067–1074.
[9] S. You, R.T. Tan, R. Kawakami, Y. Mukaigawa, K. Ikeuchi, Adherent raindrop modeling, detection and removal in video, IEEE Trans. Pattern Anal. Mach. Intell. 38 (9) (2016) 1721–1733.
[10] Y. Wang, T.-Z. Huang, X.-L. Zhao, T.-X. Jiang, Video deraining via nonlocal low-rank regularization, Appl. Math. Model. 79 (2020) 896–913.
[11] V. Santhaseelan, V.K. Asari, Utilizing local phase information to remove rain from video, Int. J. Comput. Vis. 112 (1) (2015) 71–89.
[12] X. Zhang, H. Li, Y. Qi, W.K. Leow, T.K. Ng, Rain removal in video by combining temporal and chromatic properties, in: International Conference on Multimedia and Expo, ICME, 2006, pp. 461–464.
[13] Y.-L. Chen, C.-T. Hsu, A generalized low-rank appearance model for spatio-temporally correlated rain streaks, in: IEEE International Conference on Computer Vision, ICCV, 2013, pp. 1968–1975.
[14] W. Ren, J. Tian, Z. Han, A. Chan, Y. Tang, Video desnowing and deraining based on matrix decomposition, in: IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR, 2017, pp. 2838–2847.
[15] W. Wei, L. Yi, Q. Xie, Q. Zhao, D. Meng, Z. Xu, Should we encode rain streaks in video as deterministic or stochastic? in: IEEE International Conference on Computer Vision, ICCV, 2017, pp. 2535–2544.
[16] M. Li, Q. Xie, Q. Zhao, W. Wei, S. Gu, J. Tao, D. Meng, Video rain streak removal by multiscale convolutional sparse coding, in: IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR, 2018, pp. 6644–6653.
[17] L.-J. Deng, T. Huang, X.-L. Zhao, T.-X. Jiang, A directional global sparse model for single image rain removal, Appl. Math. Model. 59 (2018) 662–679.
[18] T.-X. Jiang, T.-Z. Huang, X.-L. Zhao, L.-J. Deng, Y. Wang, A novel tensor-based video rain streaks removal approach via utilizing discriminatively intrinsic priors, in: IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR, 2017, pp. 2818–2827.
[19] T.-X. Jiang, T.-Z. Huang, X.-L. Zhao, L.-J. Deng, Y. Wang, FastDeRain: A novel video rain streak removal method using directional gradient priors, IEEE Trans. Image Process. 28 (4) (2019) 2089–2102.
[20] J.-H. Kim, J.-Y. Sim, C.-S. Kim, Video deraining and desnowing using temporal correlation and low-rank matrix completion, IEEE Trans. Image Process. 24 (9) (2015) 2658–2670.
[21] M. Li, X. Cao, Q. Zhao, L. Zhang, D. Meng, Online rain/snow removal from surveillance videos, IEEE Trans. Image Process. 30 (2021) 2029–2044.
[22] K. Zhang, W. Zuo, Y. Chen, D. Meng, L. Zhang, Beyond a Gaussian denoiser: Residual learning of deep CNN for image denoising, IEEE Trans. Image Process. 26 (7) (2017) 3142–3155.

[23] A. Kappeler, S. Yoo, Q. Dai, A.K. Katsaggelos, Video super-resolution with convolutional neural networks, IEEE Trans. Comput. Imaging 2 (2) (2016) 109–122.
[24] R. Hou, F. Li, IDPCNN: Iterative denoising and projecting CNN for MRI reconstruction, J. Comput. Appl. Math. 406 (2022) 113973.
[25] W. Yang, R.T. Tan, S. Wang, Y. Fang, J. Liu, Single image deraining: From model-based to data-driven and beyond, IEEE Trans. Pattern Anal. Mach. Intell. 43 (11) (2021) 4059–4077.
[26] W. Yang, J. Liu, J. Feng, Frame-consistent recurrent video deraining with dual-level flow, in: IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR, 2019, pp. 1661–1670.
[27] T. Liu, M. Xu, Z. Wang, Removing rain in videos: A large-scale database and a two-stream convlstm approach, in: International Conference on Multimedia and Expo, ICME, 2019, pp. 664–669.
[28] Z. Yue, J. Xie, Q. Zhao, D. Meng, Semi-supervised video deraining with dynamical rain generator, in: IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR, 2021, pp. 642–652.
[29] J. Liu, W. Yang, S. Yang, Z. Guo, D3R-Net: Dynamic routing residue recurrent network for video rain removal, IEEE Trans. Image Process. 28 (2) (2019) 699–712.
[30] X. Liu, R. Liu, L. Ma, X. Fan, Z. Luo, Spatial-temporal integration network with self-guidance for robust video deraining, in: International Conference on Multimedia and Expo, ICME, 2021.
[31] W. Zhong, X. Zhang, L. Ma, R. Liu, X. Fan, Z. Luo, Star-net: Spatial-temporal attention residual network for video deraining, in: International Conference on Multimedia and Expo, ICME, 2021.
[32] X. Xue, X. Meng, L. Ma, Y. Wang, R. Liu, X. Fan, Searching frame-recurrent attentive deformable network for real-time video deraining, in: International Conference on Multimedia and Expo, ICME, 2021.
[33] L. Ma, R. Liu, X. Zhang, W. Zhong, X. Fan, Video deraining via temporal aggregation-and-guidance, in: International Conference on Multimedia and Expo, ICME, 2021.
[34] Y.-T. Wang, X.-L. Zhao, T.-X. Jiang, L.-J. Deng, Y. Chang, T.-Z. Huang, Rain streaks removal for single image via kernel-guided convolutional neural network, IEEE Trans. Neural Netw. Learn. Syst. 32 (8) (2021) 3664–3676.
[35] C. Yu, Y. Chang, Y. Li, X. Zhao, L. Yan, Unsupervised image deraining: Optimization model driven deep CNN, in: ACM International Conference on Multimedia, ACM MM, 2021, pp. 2634–2642.
[36] L.-W. Kang, C.-W. Lin, Y.-H. Fu, Automatic single-image-based rain streaks removal via image decomposition, IEEE Trans. Image Process. 21 (4) (2012) 1742–1755.
[37] Y. Luo, Y. Xu, H. Ji, Removing rain from a single image via discriminative sparse coding, in: IEEE International Conference on Computer Vision, ICCV, 2015, pp. 3397–3405.
[38] J.-H. Kim, C. Lee, J.-Y. Sim, C.-S. Kim, Single-image deraining using an adaptive nonlocal means filter, in: International Conference on Image Processing, ICIP, 2013, pp. 914–917.
[39] H. Zhang, V.M. Patel, Convolutional sparse and low-rank coding-based rain streak removal, in: IEEE/CVF Winter Conference on Applications of Computer Vision, WACV, 2017, pp. 1259–1267.
[40] Y. Li, R.T. Tan, X. Guo, J. Lu, M.S. Brown, Rain streak removal using layer priors, in: IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR, 2016, pp. 2736–2744.
[41] S. Yan, G. Ni, T. Zeng, Image restoration based on fractional-order model with decomposition: texture and cartoon, Comput. Appl. Math. 40 (2021) 304.
[42] X. Fu, J. Huang, X. Ding, Y. Liao, J. Paisley, Clearing the skies: A deep network architecture for single-image rain removal, IEEE Trans. Image Process. 26 (6) (2017) 2944–2956.
[43] X. Fu, J. Huang, D. Zeng, Y. Huang, X. Ding, J. Paisley, Removing rain from single images via a deep detail network, in: IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR, 2017, pp. 1715–1723.
[44] S. Deng, M. Wei, J. Wang, Y. Feng, L. Liang, H. Xie, F.L. Wang, M. Wang, Detail-recovery image deraining via context aggregation networks, in: IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR, 2020, pp. 14548–14557.
[45] W. Yang, R.T. Tan, J. Feng, J. Liu, Z. Guo, S. Yan, Deep joint rain detection and removal from a single image, in: IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR, 2017, pp. 1685–1694.
[46] H. Zhang, V.M. Patel, Density-aware single image de-raining using a multi-stream dense network, in: IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR, 2018, pp. 695–704.
[47] H. Wang, Q. Xie, Q. Zhao, D. Meng, A model-driven deep neural network for single image rain removal, in: IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR, 2020, pp. 3100–3109.
[48] H. Wang, Z. Yue, Q. Xie, Q. Zhao, Y. Zheng, D. Meng, From rain generation to rain removal, in: IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR, 2021, pp. 14791–14801.
[49] R. Yasarla, V.A. Sindagi, V.M. Patel, Syn2Real transfer learning for image deraining using Gaussian processes, in: IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR, 2020, pp. 2723–2733.
[50] L. Zhu, Z. Deng, X. Hu, H. Xie, X. Xu, J. Qin, P.-A. Heng, Learning gated non-local residual for single-image rain streak removal, IEEE Trans. Circuits Syst. Video Technol. 31 (6) (2021) 2147–2159.
[51] M. Zhou, J. Xiao, Y. Chang, X. Fu, A. Liu, J. Pan, Z.-J. Zha, Image de-raining via continual learning, in: IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR, 2021, pp. 4905–4914.
[52] K. Jiang, Z. Wang, P. Yi, C. Chen, B. Huang, Y. Luo, J. Ma, J. Jiang, Multi-scale progressive fusion network for single image deraining, in: IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR, 2020, pp. 8343–8352.
[53] T. Wang, X. Yang, K. Xu, S. Chen, Q. Zhang, R.W. Lau, Spatial attentive single-image deraining with a high quality real rain dataset, in: IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR, 2019, pp. 12262–12271.
[54] K. Jiang, Z. Wang, P. Yi, C. Chen, Z. Han, T. Lu, B. Huang, J. Jiang, Decomposition makes better rain removal: An improved attention-guided deraining network, IEEE Trans. Circuits Syst. Video Technol. 31 (10) (2021) 3981–3995.
[55] H. Zhang, V. Sindagi, V.M. Patel, Image de-raining using a conditional generative adversarial network, IEEE Trans. Circuits Syst. Video Technol. 30 (11) (2020) 3943–3956.
[56] Y. Weng, G. Yang, C. Tang, H. Yang, R. Lu, F. Xu, J. Luo, iCycleGAN: An improved CycleGAN for rain streak removal from single image, in: International Conference on Industrial Artificial Intelligence, IAI, 2022, pp. 1–6.
[57] D. Ren, W. Zuo, Q. Hu, P. Zhu, D. Meng, Progressive image deraining networks: A better and simpler baseline, in: IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR, 2019, pp. 3932–3941.
[58] D. Chen, M. He, Q. Fan, J. Liao, L. Zhang, D. Hou, L. Yuan, G. Hua, Gated context aggregation network for image dehazing and deraining, in: WACV, 2019, pp. 1375–1383.
[59] J. Xiao, M. Zhou, X. Fu, A. Liu, Z.-J. Zha, Improving de-raining generalization via neural reorganization, in: IEEE International Conference on Computer Vision, ICCV, 2021, pp. 4967–4976.
[60] F. Jia, W.H. Wong, T. Zeng, DDUNet: Dense dense U-net with applications in image denoising, in: IEEE/CVF International Conference on Computer Vision Workshops, ICCVW, 2021, pp. 354–364.

[61] K. Zhang, D. Li, W. Luo, W. Ren, W. Liu, Enhanced spatio-temporal interaction learning for video deraining: A faster and better framework, IEEE Trans. Pattern Anal. Mach. Intell. (2022).
[62] P. Mu, Z. Liu, Y. Liu, R. Liu, X. Fan, Triple-level model inferred collaborative network architecture for video deraining, IEEE Trans. Image Process. 31 (2022) 239–250.
[63] W. Yang, R.T. Tan, S. Wang, J. Liu, Self-learning video rain streak removal: When cyclic consistency meets temporal correspondence, in: IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR, 2020, pp. 1717–1726.
[64] W. Yan, R.T. Tan, W. Yang, D. Dai, Self-aligned video deraining with transmission-depth consistency, in: IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR, 2021, pp. 11961–11971.
[65] J.-H. Kim, J.-Y. Sim, C.-S. Kim, Stereo video deraining and desnowing based on spatiotemporal frame warping, in: International Conference on Image Processing, ICIP, 2014, pp. 5432–5436.
[66] L. Zhu, H. Fan, Y. Luo, M. Xu, Y. Yang, Temporal cross-layer correlation mining for action recognition, IEEE Trans. Multimed. 24 (2022) 668–676.
[67] L. Zhu, Z. Xu, Y. Yang, Bidirectional multirate reconstruction for temporal modeling in videos, in: IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR, 2017, pp. 1339–1348.
[68] C. Lei, Y. Xing, Q. Chen, Blind video temporal consistency via deep video prior, in: Neural Information Processing Systems (NeurIPS), Vol. 33, 2020, pp. 1083–1093.
[69] P. Cascarano, M.C. Comes, A. Mencattini, M.C. Parrini, E.L. Piccolomini, E. Martinelli, Recursive Deep Prior Video: A super resolution algorithm for time-lapse microscopy of organ-on-chip experiments, Med. Image Anal. 72 (2021) 102124.
[70] H. Zhang, L. Mai, H. Jin, Z. Wang, N. Xu, J. Collomosse, An internal learning approach to video inpainting, in: IEEE International Conference on Computer Vision, ICCV, 2019, pp. 2720–2729.
[71] W. Yang, R.T. Tan, S. Wang, A.C. Kot, J. Liu, Learning to remove rain in video with self-supervision, IEEE Trans. Pattern Anal. Mach. Intell. (2022) http://dx.doi.org/10.1109/TPAMI.2022.3186629.
[72] J.-H. Zhuang, Y.-S. Luo, X.-L. Zhao, T.-X. Jiang, Reconciling hand-crafted and self-supervised deep priors for video directional rain streaks removal, IEEE Signal Process. Lett. 28 (2021) 2147–2151.
[73] O. Barnich, M. Van Droogenbroeck, ViBe: A universal background subtraction algorithm for video sequences, IEEE Trans. Image Process. 20 (6) (2011) 1709–1724.
[74] L. Maddalena, A. Petrosino, A self-organizing approach to background subtraction for visual surveillance applications, IEEE Trans. Image Process. 17 (7) (2008) 1168–1177.
[75] T.-X. Jiang, T.-Z. Huang, X.-L. Zhao, L.-J. Deng, Multi-dimensional imaging data recovery via minimizing the partial sum of tubal nuclear norm, J. Comput. Appl. Math. 372 (2020) 112680.
[76] A. Aravkin, S. Becker, V. Cevher, P. Olsen, A variational approach to stable principal component pursuit, in: Conference on Uncertainty in Artificial Intelligence, UAI, 2014.
[77] C. Lu, J. Feng, Y. Chen, W. Liu, Z. Lin, S. Yan, Tensor robust principal component analysis with a new tensor nuclear norm, IEEE Trans. Pattern Anal. Mach. Intell. 42 (4) (2020) 925–938.
[78] T.-X. Jiang, M.K. Ng, X.-L. Zhao, T.-Z. Huang, Framelet representation of tensor nuclear norm for third-order tensor completion, IEEE Trans. Image Process. 29 (2020) 7233–7244.
[79] V. Lempitsky, A. Vedaldi, D. Ulyanov, Deep image prior, in: IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR, 2018, pp. 9446–9454.
[80] O. Ronneberger, P. Fischer, T. Brox, U-Net: Convolutional networks for biomedical image segmentation, in: MICCAI, 2015, pp. 234–241.
[81] J.M. Bioucas-Dias, A. Plaza, N. Dobigeon, M. Parente, Q. Du, P. Gader, J. Chanussot, Hyperspectral unmixing overview: Geometrical, statistical, and sparse regression-based approaches, IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens. 5 (2) (2012) 354–379.
[82] Y.-C. Miao, X.-L. Zhao, X. Fu, J.-L. Wang, Y.-B. Zheng, Hyperspectral denoising using unsupervised disentangled spatiospectral deep priors, IEEE Trans. Geosci. Remote Sens. 60 (2022) 1–16.
[83] D. Ren, K. Zhang, Q. Wang, Q. Hu, W. Zuo, Neural blind deconvolution using deep priors, in: IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR, 2020, pp. 3338–3347.
[84] Y. Gandelsman, A. Shocher, M. Irani, ''Double-DIP'': Unsupervised image decomposition via coupled deep-image-priors, in: IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR, 2019, pp. 11018–11027.
[85] J.-H. Yang, X.-L. Zhao, T.-H. Ma, Y. Chen, T.-Z. Huang, M. Ding, Remote sensing images destriping using unidirectional hybrid total variation and nonconvex low-rank regularization, J. Comput. Appl. Math. 363 (2020) 124–144.
[86] D. Kingma, J. Ba, ADAM: A method for stochastic optimization, in: ICLR, 2014.
[87] K. Garg, S.K. Nayar, Photorealistic rendering of rain streaks, in: ACM Special Interest Group for Computer Graphics, ACM SIGGRAPH, 2006, pp. 996–1002.
[88] W. Wei, D. Meng, Q. Zhao, Z. Xu, Y. Wu, Semi-supervised transfer learning for image rain removal, in: IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR, 2019, pp. 3872–3881.
[89] K. He, G. Gkioxari, P. Dollár, R. Girshick, Mask R-CNN, in: IEEE International Conference on Computer Vision, ICCV, 2017, pp. 2980–2988.
[90] R. Barbano, J. Leuschner, M. Schmidt, A. Denker, A. Hauptmann, P. Maass, B. Jin, An educated warm start for deep image prior-based micro CT reconstruction, IEEE Trans. Comput. Imaging 8 (2022) 1210–1222.
[91] K. Zhang, M. Xie, M. Gor, Y.-T. Chen, Y. Zhou, C.A. Metzler, MetaDIP: Accelerating deep image prior with meta learning, 2022, arXiv:2209.08452.