Single Image Rain Streak Decomposition Using Layer Priors

Yu Li, *Member, IEEE,* Robby T. Tan, *Member, IEEE,* Xiaojie Guo, *Member, IEEE,*
Jiangbo Lu, *Senior Member, IEEE,* and Michael S. Brown, *Member, IEEE*

***Abstract*—Rain streaks impair visibility of an image and introduce undesirable interference that can severely affect the performance of computer vision and image analysis systems. Rain streak removal algorithms try to recover a rain streak free background scene. In this paper, we address the problem of rain streak removal from a single image by formulating it as a layer decomposition problem, with a rain streak layer superimposed on a background layer containing the true scene content. Existing decomposition methods that address this problem employ either sparse dictionary learning methods or impose a low rank structure on the appearance of the rain streaks. While these methods can improve the overall visibility, their performance can often be unsatisfactory, for they tend to either over-smooth the background images or generate images that still contain noticeable rain streaks. To address the problems, we propose a method that imposes priors for both the background and rain streak layers. These priors are based on Gaussian mixture models learned on small patches that can accommodate a variety of background appearances as well as the appearance of the rain streaks. Moreover, we introduce a structure residue recovery step to further separate the background residues and improve the decomposition quality. Quantitative evaluation shows our method outperforms existing methods by a large margin. We overview our method and demonstrate its effectiveness over prior work on a number of examples.**

***Index Terms*—Rain removal, image decomposition, visibility recovery, image prior, Gaussian mixture models.**

I. Introduction

MOST computer vision and image processing algorithms assume that the input image is of scene content that is clear and visible. However, for outdoor images, undesirable interference from rainy weather is often inevitable. Rain introduces several different types of visibility degradation. Raindrops that fall and flow on a camera lens or a windscreen can obstruct, deform, and/or blur the imagery of the background scenes. Distant rain streaks accumulated throughout the scene reduce the visibility in a manner similar to fog–namely by scattering light out and into the line of sight, creating a veiling phenomenon. Nearby rain streaks, where the individual rain streaks are visible, can also significantly degrade visibility due to their specular highlights, scattering, and blurring effect. Figure 1 shows an example of visibility degradation due to rain streaks and our attempt to remove them.

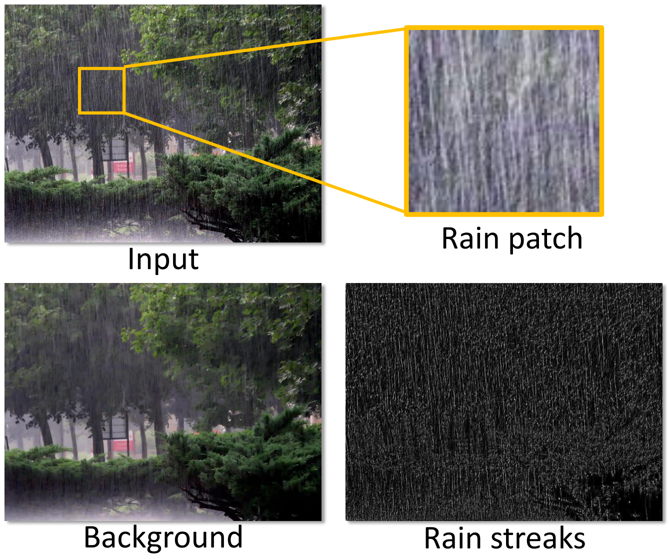


Fig. 1: **Upper Row:** An example rain image and a zoomed-in patch that mainly contains an annoying effect of rain streaks. Our method learns a rain streak layer prior on this region and used it for layer separation. **Lower Row:** The background and rain streak layers recovered by our proposed method, respectively. Note the rain streak layer is amplified for better visualization.

Mathematically, the observed rain image $\mathbf{O} \in \mathbb{R}^{M\times N}$ can be modeled as a linear superimposition [1], [2] of the desired background layer $\mathbf{B} \in \mathbb{R}^{M\times N}$ and the rain streak layer $\mathbf{R} \in \mathbb{R}^{M\times N}$, expressed as $\mathbf{O} = \mathbf{B}+\mathbf{R}$. The goal of rain streak removal is to decompose the rain-free background $\mathbf{B}$ and the rain streak layer $\mathbf{R}$ from an input image $\mathbf{O}$, and hence enhance the visibility of the image. This layer separation problem is ill-posed, as the number of unknowns to be recovered is twice as many as that of the input. One common strategy is to use multiple images, or a video sequence, to mitigate the difficulty of the background scene recovery given the rich temporal

This study is supported by the HCCS research grant from A*STAR. R. T. Tan's work in this research is supported by the National Research Foundation, Prime Ministers Office, Singapore under its International Research Centre in Singapore Funding Initiative. X. Guo is supported by National Natural Science Foundation of China (grant no. 61402467). This research was undertaken thanks in part to funding from the Canada First Research Excellence Fund for the Vision: Science to Applications (VISTA) programme.

Y. Li is with the Advanced Digital Sciences Center, Singapore (e-mail: ian.li.liyu@gmail.com).
R. T. Tan is with Yale-NUS College and National University of Singapore (email:tanrobby@gmail.com).
X. Guo is with State Key Lab of Information Security, IIE, CAS, China (email:guoxiaojie@iie.ac.cn).
J. Lu is with the Shenzhen Cloudream Technology Co., Ltd., Shenzhen, China and also holds an adjunct position with the Advanced Digital Sciences Center, Singapore (e-mail: jiangbo.lu@gmail.com).
M. S. Brown is with York University, Canada (e-mail: mbrown@cse.yorku.ca).
This work was partly done when J. Lu was working in ADSC.

information. In this paper, however, we focus on the problem of rain streak removal given a single image only.

While removing rain streaks in a single image is challenging as less information is available, single image approaches are desired for two main reasons. First, in many situations, the input is only a single image captured in a rainy environment (e.g., archived images, images available on the Internet, images taken by still cameras). Second, for dynamic scenes, such as when the camera moves and some part of the scene also moves, temporal information might not be reliable. As a result, being able to remove rain streaks for each frame will benefit the overall final result, since most existing video-based methods assume a static background.

To make the problem well-posed and tractable, we enforce layer priors on both the background and rain components. More specifically, the idea of using patch-based priors is inspired by Zoran and Weiss [3], who used Gaussian mixture models (GMMs) to model natural image patches. This approach is simpler to compute than the existing prior models, such as FoE [4] or Weiss-Freeman's priors [5]. Recent works from Shih et al. [6] shows the superiority of using GMMs as priors in solving the reflection removal problems [7].

In our rain removal approach, to model the background patch priors, we use a GMM trained on patches from natural images. An additional gradient sparsity constraint is imposed to further regularize the background. As for the rain layer, we do the same: we train a GMM from small image patches of rain streaks. The difference in rain GMM learning is that we use only the input image and select a region with textureless background to generate the rain patches. Unlike existing methods in single-image rain streak removal ([2], [1], [8]), our method is straightforward and generates considerably better results qualitatively and quantitatively. To our knowledge, this is the first method to use GMM patch priors for the purpose of rain streak removal.

A shorter version of this work appeared in [9]. This journal version presents the algorithm in more technical details and includes a new structure residue recovery step to recover the additional background residue in the rain streak output from our optimization. This step can further improve layer separation quality over our original method in [9]. In addition, we evaluate the proposed method on a synthetic data set by measuring the quality of both the recovered background and the rain streak layer. We also add a comparison to a recent work of [8]. Experiments show that our method, particularly with the structure residue recovery step, can outperform the existing methods by a large margin in terms of both SSIM and PSNR metrics.

The remainder of this paper is organized as follows. Section II discusses the related methods that deal with rain, including video-based rain streak removal and single image-based rain streak removal. Section III details our method, including the problem formulation and optimization. Section IV shows the results and analyzes them in comparison with the results of other methods. The paper is concluded in Section V.

II. Related work

There are a number of methods proposed to improve the visibility of images captured in bad weather like haze and fog (e.g. [10], [11], [12]), rain, or snow (e.g., [13], [1], [2]). Methods specifically dealing with rain streak interference fall into two categories: multiple image/video-based and single image methods.

A. Video-Based Methods

Early methods to remove rain streaks include work by Garg and Nayar [14], [13], which introduces a rain streak detection and removal method from a video sequence. The detection is based on two constraints: first, since rain streaks are dynamic, their changes in intensity within a few frames are relatively large. Second, since other objects are also possibly dynamic, rain streaks can be differentiated from these objects by verifying whether the intensity changes along the streak are photometrically linearly related to the background intensity. This second constraint will reduce the false alarms introduced by the first constraint. Having detected the rain streaks, the method removes them by taking the average intensity of the pixels taken from the previous and subsequent frames.

Garg and Nayar [15] propose another method that exploits the ability to control a video camera's operational parameters when capturing a rainy scene. To this end, they show that rain visibility in images relies significantly on the exposure time and depth of field of the camera. Thus adjusting these parameters while taking the video will allow us to reduce the appearance of the rain streaks. Zhang et al. [16] added an additional constraint called the chromaticity constraint. They found the intensity changes in the RGB color channels are the same for pixels representing rain streaks.

More recently, Bossu et al. [17] propose a rain detection algorithm based on the histogram of streak orientations. The main idea is to fit a Gaussian distribution on rain streak histograms, such that rain streaks can be differentiated from other dynamic objects. It uses background subtraction to obtain the rain histograms. Barnum et al. [18] develop a model of the rain profiles in a frequency domain and analyze the properties of rain in the domain. Their frequency analysis allows for detection and removal of rain or snow streaks. Santhaseelan and Asari [19] apply phase congruency features to detect rain streaks, and then reconstruct the background scene using the registration of phase information obtained from optical flow (to account for moving objects present in the scene). There are also methods that treat rain removal in video as a low-rank tensor completion problem–for example, [20], [21], assuming a constant background (see [22] for a complete review on the existing video-based rain streak removal).

B. Single Image Methods

For single-image rain streak removal, Kang *et al.* [1] propose a method that decomposed an input image into its low-frequency component (the structure layer) and a high-frequency component (the texture layer). The high-frequency part contains rain streaks and edges of background objects. They separate the rain streak frequency from the high-frequency layer via sparse coding-based dictionary learning with HoG features. The output is obtained by combining back the low-frequency and processed high-frequency layers. While the decomposition idea is elegant, the overall framework in [1] is complex. The results are not optimal either, particularly as the background tends to be blurry, which is caused by too few high-frequency component in the background found using the dictionary. These problems remain in the follow-up works of this method–for example, [23], [24].

More recently, Chen and Hsu [2] introduce a single objective function to decompose the background and rain streak layers. They formulate a cost function with three terms: the likelihood, a smoothed background layer, and a low-rank rain streak layer. Although the idea of posing the problem into an objective function is attractive, the constraints seem not sufficiently strong. In our experiments, we still observe a large amount of rain streaks in the output. Kim et al. [25] detect rain streaks by assuming the elliptical shape and the vertical orientation of the rain, and remove the detected streaks using nonlocal mean filtering. This idea works for some cases of rain streaks, but unfortunately detecting individual streaks is challenging, because they can possibly have different orientations, scales, and densities. The most recent work of [8] proposes a discriminative sparse coding framework to remove rain streaks in a single image. While effective, the approach often still leaves noticeable thin structures from the rain streaks in the final output.

Aside from dealing with rain streaks, several methods have been proposed to address artifacts that arise when raindrops adhered to the camera lens or a windscreen in front of the camera (e.g., [26], [27], [28], [29], [30]). The problems specific to adherent raindrops, however, are notably different from the interference caused by rain streaks. In particular, static adherent raindrops, which is the problem most existing methods target, are generally less dense than rain streaks. In addition, their size in an image is generally much larger than rain streaks, and they tend to completely occlude parts of the background scene.

III. Our Method

A. Problem Formulation

As introduced, the observed rain image $\mathbf{O} \in \mathbb{R}^{M\times N}$ can be modeled as a linear superimposition of the desired background layer $\mathbf{B} \in \mathbb{R}^{M\times N}$ and the rain streak layer $\mathbf{R} \in \mathbb{R}^{M\times N}$, such that:

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

Hence, the goal of rain streak removal is to decompose the rain-free background $\mathbf{B}$ and the rain streak layer $\mathbf{R}$ from a given input image $\mathbf{O}$. As previously stated, this problem is ill-posed. To resolve it, we propose to maximize the joint probability of the background layer and the rain layer using the MAP (maximum a posteriori): that is, maximize $p(\mathbf{B},\mathbf{R}|\mathbf{O}) \propto p(\mathbf{O}|\mathbf{B},\mathbf{R})\cdot p(\mathbf{B})\cdot p(\mathbf{R})$ with the assumption that the two layers $\mathbf{B}$ and $\mathbf{R}$ are independent. Equivalently, with slight algebraic manipulation on its negative log function, we obtain the following energy minimization problem:

$$\begin{aligned}\min_{\mathbf{B},\mathbf{R}} \quad & \|\mathbf{O}-\mathbf{B}-\mathbf{R}\|_F^2 + \Phi(\mathbf{B}) + \Psi(\mathbf{R}) \\ \text{s.t.} \quad & \forall i \;\; 0 \le \mathbf{B}_i, \mathbf{R}_i \le \mathbf{O}_i,\end{aligned} \tag{2}$$

where $\|\cdot\|_F$ represents the Frobenius norm and i is the pixel index. The first term $\|\mathbf{O}-\mathbf{B}-\mathbf{R}\|_F^2$ is essentially concerned with the fidelity between the observed and the recovered signals, while $\Phi(\mathbf{B})$ and $\Psi(\mathbf{R})$ designate the priors that will be respectively imposed on $\mathbf{B}$ and $\mathbf{R}$ to regularize the inference. The inequality constraint ensures that the desired $\mathbf{B}$ and $\mathbf{R}$ are positive images. More importantly, this inequality constraint plays a critical role in estimating reliable solutions as it regularizes the DC component of the recovered layers as verified in recent works by [31], [6].

Focusing our discussion on the priors, we first define the priors of the background layer as:

$$\Phi(\mathbf{B}) := -\gamma \sum_i \log \mathcal{G}_{\mathbf{B}}(\mathcal{P}(\mathbf{B}_i)) + \alpha\|\nabla\mathbf{B}\|_1, \tag{3}$$

where $\gamma(=0.01)$ and $\alpha(=0.05)$ are two non-negative coefficients balancing the corresponding terms. The function $\mathcal{P}(\mathbf{B}_i)$ is to extract the $n\times n$ (pre-defined size) patch around pixel $\mathbf{B}_i$ and reshape it into a vector of length n^2 with the DC component removed. The term $\mathcal{G}(\mathbf{x})$ stands for the GMM of $\mathbf{x}$–that is, $\mathcal{G}(\mathbf{x}) := \sum_{k=1}^{K}\pi_k\mathcal{N}(\mathbf{x}|\mu_k,\mathbf{\Sigma}_k)$, where K is the total number of Gaussian components, π_k is the component weight such that $\sum_{k=1}^{K}\pi_k = 1$, while μ_k and $\mathbf{\Sigma}_k$ are the mean and covariance corresponding to the kth component, respectively. As the function $\mathcal{P}(\cdot)$ has removed the mean of every patch, $\mu_k = \mathbf{0}$ for all k. The benefit of this patch regularizer based on GMM has been demonstrated in [3]. In addition, it has been widely recognized that natural images are largely piecewise smooth and their gradient fields are typically sparse. Therefore we employ $\|\nabla\mathbf{B}\|_1$ to achieve such a functional, where ∇ denotes the gradient operator and $\|\cdot\|_1$ is the ℓ^1 norm.

As for the priors of the rain layer, we write it in the following form:

$$\Psi(\mathbf{R}) := -\gamma \sum_i \log \mathcal{G}_{\mathbf{R}}(\mathcal{P}(\mathbf{R}_i)) + \beta\|\mathbf{R}\|_F^2, \tag{4}$$

where $\mathcal{G}_{\mathbf{R}}(\mathcal{P}(\mathbf{R}_i))$ is similar with $\mathcal{G}_{\mathbf{B}}(\mathcal{P}(\mathbf{B}_i))$. Note that we use two different GMMs for the background and rain layers, termed $\mathcal{G}_{\mathbf{B}}$, and $\mathcal{G}_{\mathbf{R}}$, respectively (we discuss further this GMM modeling in Section III-C). Since the rain component tends to make up a small fraction of the observation, we impose $\|\mathbf{R}\|_F^2$ to penalize it, the importance of which is controlled by the parameter $\beta(=0.01)$.

Putting all terms together leads to the complete formulation of the energy function:

$$\begin{aligned}\min_{\mathbf{B},\mathbf{R}} \quad & \|\mathbf{O}-\mathbf{B}-\mathbf{R}\|_F^2 + \alpha\|\nabla\mathbf{B}\|_1 + \beta\|\mathbf{R}\|_F^2 - \\ & \gamma\sum_i \log\left(\mathcal{G}_{\mathbf{B}}(\mathcal{P}(\mathbf{B}_i)) + \log\mathcal{G}_{\mathbf{R}}(\mathcal{P}(\mathbf{R}_i))\right) \\ \text{s.t.} \quad & \forall i \;\; 0 \le \mathbf{B}_i, \mathbf{R}_i \le \mathbf{O}_i.\end{aligned} \tag{5}$$

The optimization approach to minimize this energy function is discussed in the next section.

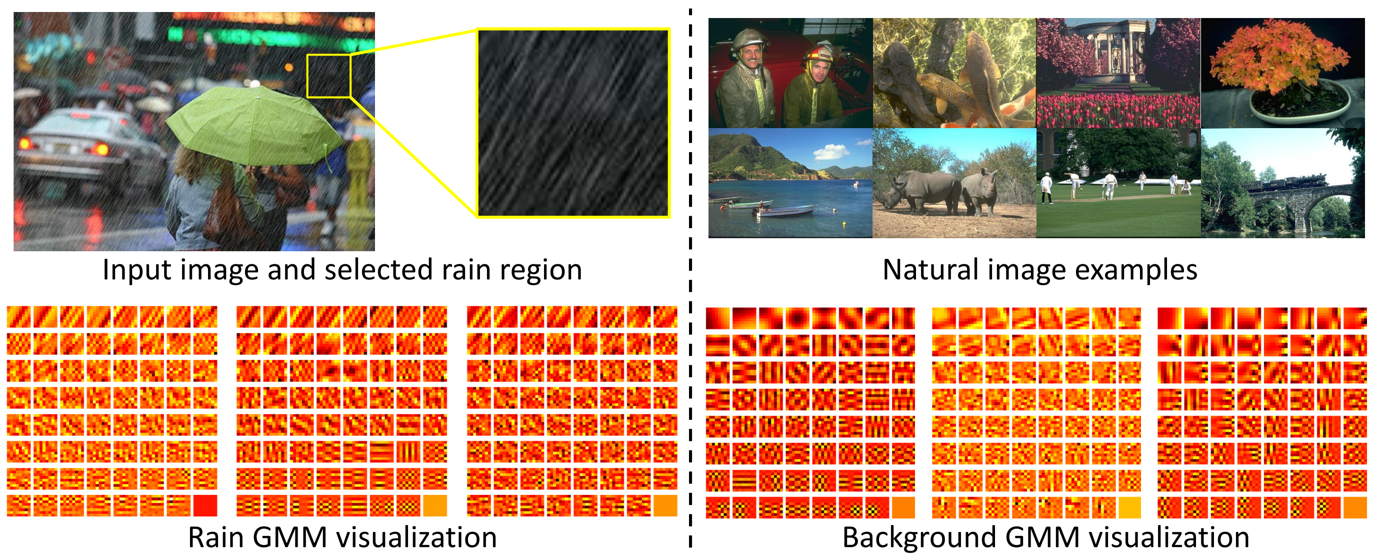

Input image and selected rain region
Natural image examples
Rain GMM visualization
Background GMM visualization

Fig. 2: (Upper left) The input image and the selected region used to learn the rain streak GMM. (Bottom left) Visualization of the eigenvectors of covariance of three randomly picked rain GMM components, sorted by their eigenvalues in descending order. The visualization helps to reveal that the GMM can capture the rain orientation and structure information. (Right) Some example natural images and the visualization of the eigenvectors of covariance of three randomly picked GMM components.

B. Optimization

As noticed in Eq. (5), the cost function is non-convex due to the patch GMM priors. A commonly used scheme to solve this kind of problem is the half-quadratic splitting technique [32]. To cast our problem into the half-quadratic splitting framework, we need to make the objective function separable. Hence, auxiliary variables $\mathbf{g}_{\mathbf{B}_i}$, $\mathbf{g}_{\mathbf{R}_i}$ and $\mathbf{H}$ are introduced to replace $\mathcal{P}(\mathbf{B}_i)$, $\mathcal{P}(\mathbf{R}_i)$ and $\nabla\mathbf{B}$, respectively. By doing so, the optimization problem turns out to be in the following form:

$$\begin{aligned}\min\ & \|\mathbf{O}-\mathbf{B}-\mathbf{R}\|_F^2+\alpha\|\mathbf{H}\|_1+\beta\|\mathbf{R}\|_F^2- \\ & \gamma\sum_i\big(\log\mathcal{G}_{\mathbf{B}}(\mathbf{g}_{\mathbf{B}_i})+\log\mathcal{G}_{\mathbf{R}}(\mathbf{g}_{\mathbf{R}_i})\big)+\omega\|\nabla\mathbf{B}-\mathbf{H}\|_F^2 \\ & +\omega\sum_i\big(\|\mathcal{P}(\mathbf{B}_i)-\mathbf{g}_{\mathbf{B}_i}\|_2^2+\|\mathcal{P}(\mathbf{R}_i)-\mathbf{g}_{\mathbf{R}_i}\|_2^2\big) \\ & \text{s.t.}\quad \forall i\ \ 0\le\mathbf{B}_i,\mathbf{R}_i\le\mathbf{O}_i,\end{aligned}\tag{6}$$

where $\|\cdot\|_2$ represents the ℓ^2 norm. Notice that ω is a positive parameter that monotonically increases after each iteration (starting from $\omega_0=0.02$). As ω grows, the solutions to Eq. (6) infinitely approach those to Eq. (5). The proposed algorithm iteratively updates the variables as described in the following.

a) Solving $\mathbf{H}$*:* Discarding the variables unrelated to $\mathbf{H}$ yields:

$$\mathbf{H}^{(t+1)}=\operatorname*{argmin}_{\mathbf{H}}\alpha\|\mathbf{H}\|_1+\omega\|\nabla\mathbf{B}^{(t)}-\mathbf{H}\|_F^2.\tag{7}$$

This is a classic LASSO problem. Its closed-form solution can be efficiently obtained by the shrinkage operator, the definition of which on scalars is $\mathcal{S}_{\epsilon>0}[x]:=\operatorname{sgn}(x)\max(|x|-\epsilon,0)$. The extension of the shrinkage operator to vectors and matrices is simply applied element-wise. As a result, we have:

$$\mathbf{H}^{(t+1)}=\mathcal{S}_{\alpha/2\omega}[\nabla\mathbf{B}^{(t)}].\tag{8}$$

b) Solving $\{\mathbf{B},\ \mathbf{R}\}$*:* By fixing $\mathbf{H}$, $\mathbf{g}_{\mathbf{B}_i}$, and $\mathbf{g}_{\mathbf{R}_i}$, the optimization problem corresponding to $\{\mathbf{B},\ \mathbf{R}\}$ becomes:

$$\begin{aligned}\{\mathbf{B}^{(t+1)},\mathbf{R}^{(t+1)}\}=&\operatorname*{argmin}_{\mathbf{B},\mathbf{R}}\|\mathbf{O}-\mathbf{B}-\mathbf{R}\|_F^2+\beta\|\mathbf{R}\|_F^2+ \\ &\omega\sum_i\big(\|\mathcal{P}(\mathbf{B}_i)-\mathbf{g}_{\mathbf{B}_i}^{(t)}\|_2^2+\|\mathcal{P}(\mathbf{R}_i)-\mathbf{g}_{\mathbf{R}_i}^{(t)}\|_2^2\big) \\ &\text{s.t.}\quad\forall i\ \ 0\le\mathbf{B}_i,\mathbf{R}_i\le\mathbf{O}_i.\end{aligned}\tag{9}$$

Following [6], we use L-BFGS [33] to minimize this constrained L_2 problem. L-BFGS is an effective solver for this problem, since the involved terms in this problem are all quadratic.

c) Solving $\mathbf{g}_{\mathbf{B}_i}$ *(*$\mathbf{g}_{\mathbf{R}_i}$*):* Since the $\mathbf{g}_{\mathbf{B}_i}$ and $\mathbf{g}_{\mathbf{R}_i}$ sub-problems share the same formulation with the other variables given, we detail only the solver of $\mathbf{g}_{\mathbf{B}_i}$ here, while $\mathbf{g}_{\mathbf{R}_i}$ can be updated analogously. The optimization problem associated with each $\mathbf{g}_{\mathbf{B}_i}^{(t+1)}$ is expressed as:

$$\mathbf{g}_{\mathbf{B}_i}^{(t+1)}=\operatorname*{argmin}_{\mathbf{g}_{\mathbf{B}_i}}\omega\|\mathcal{P}(\mathbf{B}_i^{(t+1)})-\mathbf{g}_{\mathbf{B}_i}\|_2^2-\gamma\log\mathcal{G}_{\mathbf{B}}(\mathbf{g}_{\mathbf{B}_i}).\tag{10}$$

The approximate optimization suggested by [3] is used. This approach applies Wiener filtering using only the component with the largest likelihood in the GMM. The whole process is summarized in Algorithm 1.

Note that we first convert the RGB input image to the YUV space and remove rain streaks on the luminance (Y) channel. We get the final output by converting the rain streak removal result in YUV back to RGB space.

C. Gaussian Mixture Model Learning

In this section, we describe how to obtain the two GMMs for the background and rain layers– namely $\mathcal{G}_{\mathbf{B}}$ and $\mathcal{G}_{\mathbf{R}}$. To obtain $\mathcal{G}_{\mathbf{B}}$, we utilize a pre-trained GMM model provided by

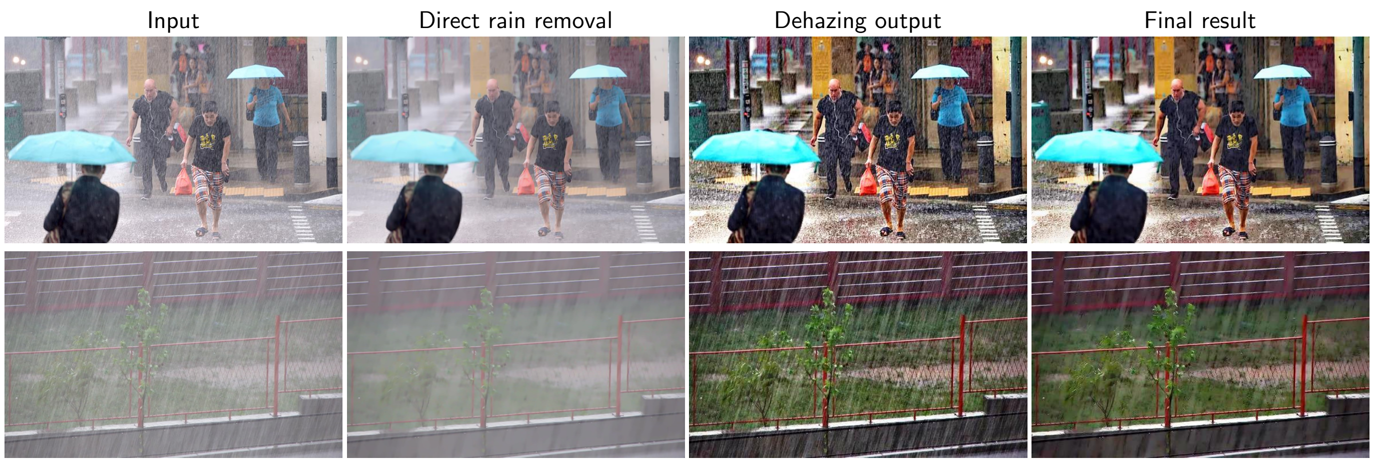


Fig. 3: Rain streak removal example for heavy rain. We found that first applying a dehazing method (3rd column), followed by the rain streak removal, helps improve results.

Algorithm 1 Rain Streak Removal Using Layer Priors

```
Input: input image $\mathbf{O}$; GMMs for two layers: $\mathcal{G}_\mathbf{B}$ and $\mathcal{G}_\mathbf{R}$;
Initialization: $\mathbf{B} \leftarrow \mathbf{O}$; $\mathbf{R} \leftarrow 0$; $\omega \leftarrow \omega_o$;
  repeat
    update $\mathbf{H}$ using Eq. (8);
    solve $\{\mathbf{B}, \mathbf{R}\}$ by Eq. (9);
    solve $\{\mathbf{g}_{\mathbf{B}_i}, \mathbf{g}_{\mathbf{R}_i}\}$ by Eq. (10);
    $\omega = 2 * \omega$;
  until convergence or maximum iteration number;
Output: The estimation of two layers $\mathbf{B}$ and $\mathbf{R}$;
```

[3] with 200 mixture components and patch size 8×8. This model is learned using Expectation–Maximization (EM) from a set of 2×10^6 patches sampled from 300 natural images with their DC removed. The images are all rain streak-free images and thus can directly serve our purpose of modeling the background layer.

To obtain $\mathcal{G}_\mathbf{R}$, existing methods attempt to extract the internal properties of rain streaks within the input image itself, like [2], [1]. This means we do not require building a dataset of rain streaks to learn the rain streak model. Instead, we directly learn the priors from the specific input image. Doing this guarantees the correctness of the rain streak appearance, which otherwise can be considerably different from one image to another image. Unlike [2], [1], which work on the entire image, we found that $\mathcal{G}_\mathbf{R}$ requires only small regions (e.g. , a region with size 100×100 contains about $10K$ overlapping 8×8 patches inside), since rain streaks mostly form repetitive patterns.

We observe that most natural images contain regions that are relatively homogenous background–for instance, sky and walls. The image patches within these regions can be treated as pure rain streaks, and used to train $\mathcal{G}_\mathbf{R}$. To select such regions, we calculate the variance within every possible region in a sliding window fashion and pick the ones with the least variances. We further remove uniformly bright regions where rain streaks are hardly observed by removing the candidates with mean intensity higher than a threshold (0.8). This strategy of using local patches for global image restoration shares the similar spirit in [34].

After detecting the pure rain streak region (e.g., see the rectangular box in Figure 1), we samples 8×8 small patches from this region and perform EM to learn the parameters of $\mathcal{G}_\mathbf{R}$. We set the cluster number for $\mathcal{G}_\mathbf{R}$ to a small one–that is, 20–compared with 200 for $\mathcal{G}_\mathbf{B}$, as the rain streak appearance of one single image has less variance than the background layer.

Figure 2 (left) shows the eigenvectors of the three randomly selected mixture components from the learned $\mathcal{G}_\mathbf{R}$. Note that they have rain streak structures that contribute much to the expressive power of the model for the rain streak layer. On the right side of Figure 2, we also visualize the eigenvectors of the three randomly selected mixture components from $\mathcal{G}_\mathbf{B}$, learned on natural image patches. As can be seen, the components in $\mathcal{G}_\mathbf{B}$ show clear differences with those in $\mathcal{G}_\mathbf{R}$, as they contain more diversity.

D. Dealing with Heavy Rain

For images taken in scenes with heavy rainfall, the density of the rain streaks accumulated throughout the environment and partially occludes the scene content. As a result, the rain streaks scatter the atmosphere light and produce a 'washed-out' effect similar to haze and fog [11]. In such cases, the individual streaks, mainly nearby rain streaks, cannot be observed anymore. For this case, we found that applying a dehazing method–for example, [35]–as a preprocessing step can be useful. Two examples are shown in Figure 3. We can see the individual rain streaks are clearer after the dehazing preprocessing step. Having applied the dehazing, we then run our rain streak removal method, which produces clearer results in terms of visibility, as shown in Figure 3. This preprocessing step could be an option to the users when they want to handle heavy rain cases.

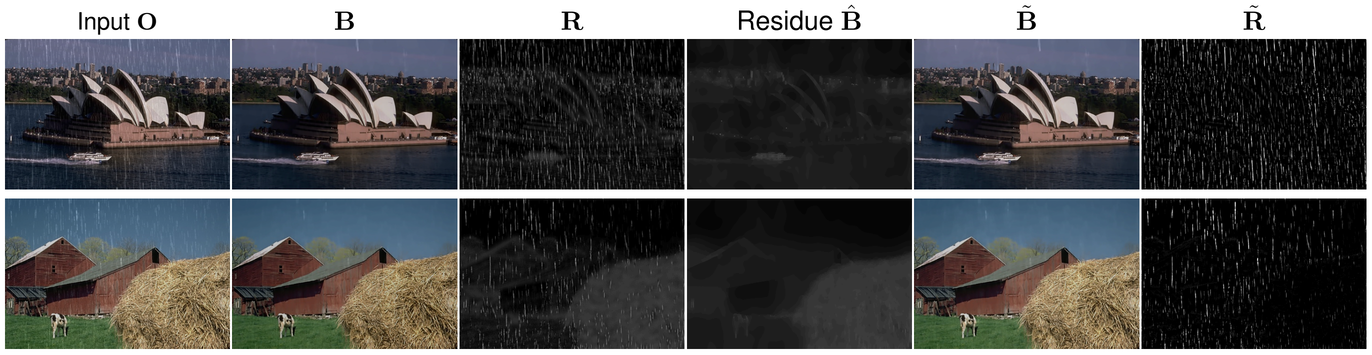


Fig. 4: Background-structure-residue recovery illustration. We can effectively retrieve the background structure residues $\hat{\mathbf{B}}$ in the initial recovery of the background. Finding these residues helps separate the two layers, further as shown in $\tilde{\mathbf{B}}$ and $\tilde{\mathbf{R}}$. (Rain streak and structure residue layers are amplified for better visualization.)

E. Recovering the Background Structure Residue

While our proposed layer decomposition method can effectively separate most background components from rain streak, a closer look at our rain streak layer may still reveal weak background structure residues in some cases, as shown in Figure 4. These background structure residues, which should belong to the background layer, are falsely assigned to the rain streak layer in our optimization. To address this problem, we introduce a refinement step to further retrieve these background structure residues from the rain streak layer.

To achieve this, we use the background output $\mathbf{B}$ after the optimization as guidance to further clean the rain streak output $\mathbf{R}$, using the guided image smoothing framework [36], [37]. Specifically, we attempt to obtain the background structure residues $\hat{\mathbf{B}}$ by optimizing the following objective function:

$$\min_{\hat{\mathbf{B}}} \sum_i (\hat{\mathbf{B}}_i - \mathbf{R}_i)^2 + \tau \sum_i \sum_{j \in \mathcal{N}_4(i)} \phi_{i,j}(\mathbf{B})(\hat{\mathbf{B}}_i - \hat{\mathbf{B}}_j)^2 \quad (11)$$

where i is the pixel index. $\mathcal{N}_4(i)$ represents the four neighbors for pixel i. The spatially varying weight function $\phi_{i,j}$ measures how similar two pixels i and j are at the guidance image $\mathbf{B}$. The first term is the data constraint, which forces the closeness of residue $\hat{\mathbf{B}}$ to $\mathbf{R}$, while the second term encourages the smoothness under the guidance of $\mathbf{B}$. The weight τ balances the data term and the regularization term. We use the fast guided image smoothing techniques [37] to solve Eq. (11).

Having obtained the background structure residues $\hat{\mathbf{B}}$, we remove it from the rain streak layer $\mathbf{R}$ and add it back to the background layer $\mathbf{B}$. The final background layer and rain streak layer are denoted as $\tilde{\mathbf{B}}$ and $\tilde{\mathbf{R}}$ respectively and can be computed as

$$\begin{aligned} \tilde{\mathbf{R}} &= \max(\mathbf{R} - \hat{\mathbf{B}}, 0); \\ \tilde{\mathbf{B}} &= \mathbf{B} + \hat{\mathbf{B}}. \end{aligned} \quad (12)$$

Figure 4 shows the results of this background structure residue recovery step. As can be seen, this step can more successfully recover the background residues and make the rain streak cleaner. An improvement in the Structure Similarity Index (SSIM) is also demonstrated.

IV. Experimental Results

We evaluate our method using both synthetic and real images, and compare our results with the state-of-the-art methods, including the sparse representation-based dictionary learning method [1] (denoted as SR), the low-rank appearance method [2] (denoted as LRA), and the discriminative sparse coding approach [8] (denoted as DSC). For SR and DSC, we use the codes provided by the author with their default parameter settings. We have implemented LRA by strictly following the procedure described in [2] and the parameters are fixed to get the best overall performance in the quantitative evaluation. For the quantitative experiments on synthetic data, the ground truth images are available, and we can evaluate and compare the results using SSIM [38] as well as PSNR on the luminance channel (in both two metrics, a larger number indicates a closer proximity to the ground truth).

Our Matlab implementation takes $93s$ (5 iterations) / $370s$ (20 iterations) to process one 480×640 color image on a PC with Intel I7 CPU (3.4GHz) and 8GB RAM, and most of the time is spent on solving Eq. (9) using L-BFGS. Note that in all the results presented in the section, we do not apply the dehazing processing as described in Section III-D for fair comparison with other methods.

A. Synthetic Data

1) Study on parameters: First we present a quick study on the convergence and parameter settings of our method. We tested the effect on the SSIM of recovered layers to the ground truth on our synthetic dataset and plotted in Figure 7. It can be seen from Figure 7(a) that our method converges fast such that it can get to a good solution after about 10 iterations. Figure 7(b)(c)(d) shows the layer recovery results on different parameter setting of α, β, and γ. Our method is not sensitive to the parameter settings and is quite stable when the parameters are within a proper range.

From the objective function in Eq. (5), one can notice that by simply disabling the terms related to $\mathbf{g}_{\mathbf{B}_i}$ and $\mathbf{g}_{\mathbf{R}_i}$ (set $\gamma = 0$), our model becomes dominated by the gradient sparsity term $\alpha \|\nabla \mathbf{B}\|_1$, which is the Total Variation (TV) model [39]. To reveal the benefit of the GMM priors, a comparison

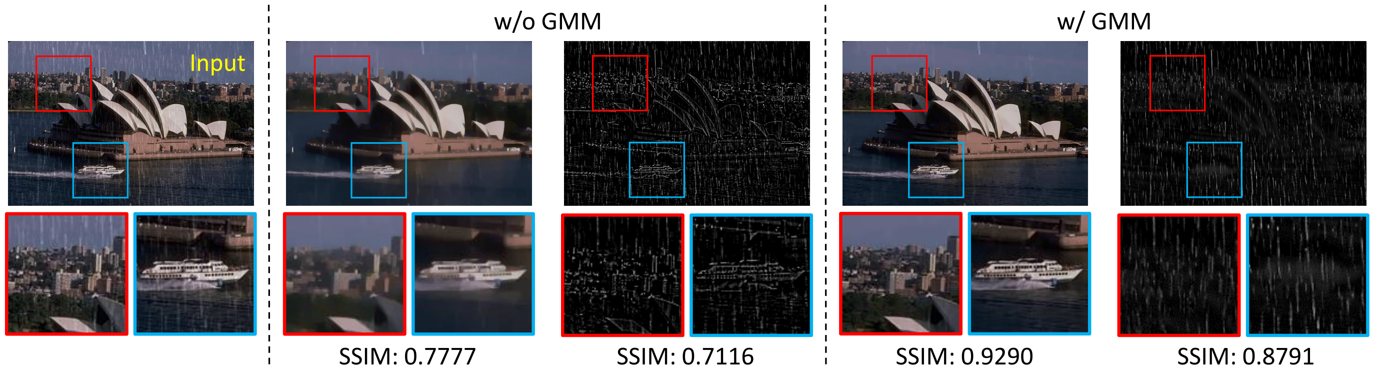


Fig. 5: Illustration of the effect of the GMM. Our objective without the GMM components (second column) cannot distinguish the rain streaks and the image details like the one with GMM (left two columns). The Structure Similarity Index (SSIM) for each result is shown below the image.

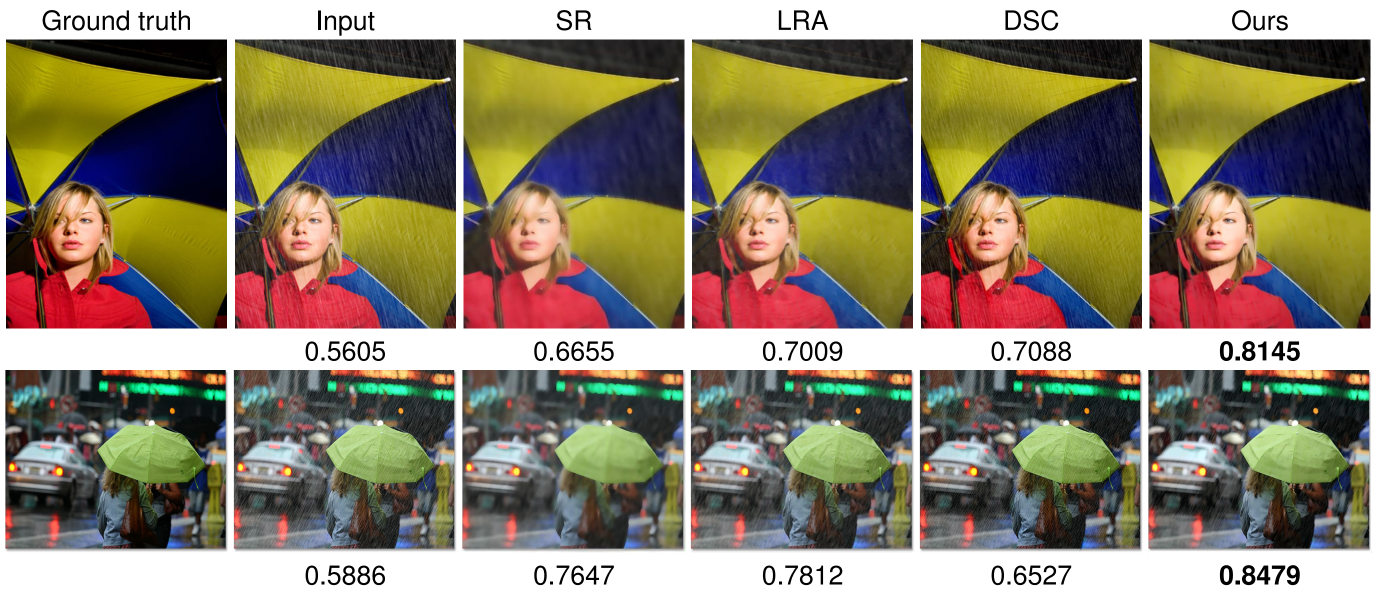


Fig. 6: Rain streak removal results for the two data sets in [1]. The numbers below images are the SSIM wrt ground truth. Ours shows better background recovery both quantitatively and qualitatively.

between the result with and without the GMM is conducted. The effect is shown in Figure 5, from which, we can see that the result without GMM part is able to filter out the rain streaks but also falsely removes many details that belong to the background as it will have the behavior similar to the of TV filter. The GMM term greatly helps distinguish the rain streaks from other scene details in the recovering rain layers. The result of SSIM score increases from (0.7777, 0.7116) without the GMM to (0.9290, 0.8791) with the GMM. This increase demonstrates the effectiveness of the GMM model.

2) Comparison: We compare our method with SR [1], LRA [2], and DSC [8]. Figure 6 shows the results of these methods on the data set provided in [1]. This is the only dataset we can find that is provided by previous work and has ground truth for quantitative evaluation. As observed, our method considerably outperforms the other three methods in terms of both visual quality and SSIM. SR [1] tends to over-smooth the image content, LRA [2] sometimes fails to capture the rain streaks, and DSC [8] leaves a noticeable amount of rain in the outputs. Our proposed method removes the rain streaks while keeping image details in the background layer.

For more comparison, we synthesize a new data set with 12 images using the photorealistic rendering techniques proposed by Grag and Shree [41]. The background images are from the BSD300 Dataset [42], which are all natural images. Table I lists the performance of the methods. Our method with the background structure residue recovery step described in Section III-E is denoted as 'Ours+BSR,' while our original method is denoted as 'Ours.' We have also added a generic edge-aware smoothing filter, the guided filter (GF) [40], here for comparison. The numerical performance using SSIM and PSNR metrics of different methods is listed in Table I. Figure 8 shows the visual results of two examples.

We first compare the background recovery quality. It is expected that the performance of the generic filtering method (GF) is inferior to those task-specific rain streak removal methods. It is surprising to find that the background recovery of GF has an even higher SSIM and PSNR (0.8179 / 29.43 dB)

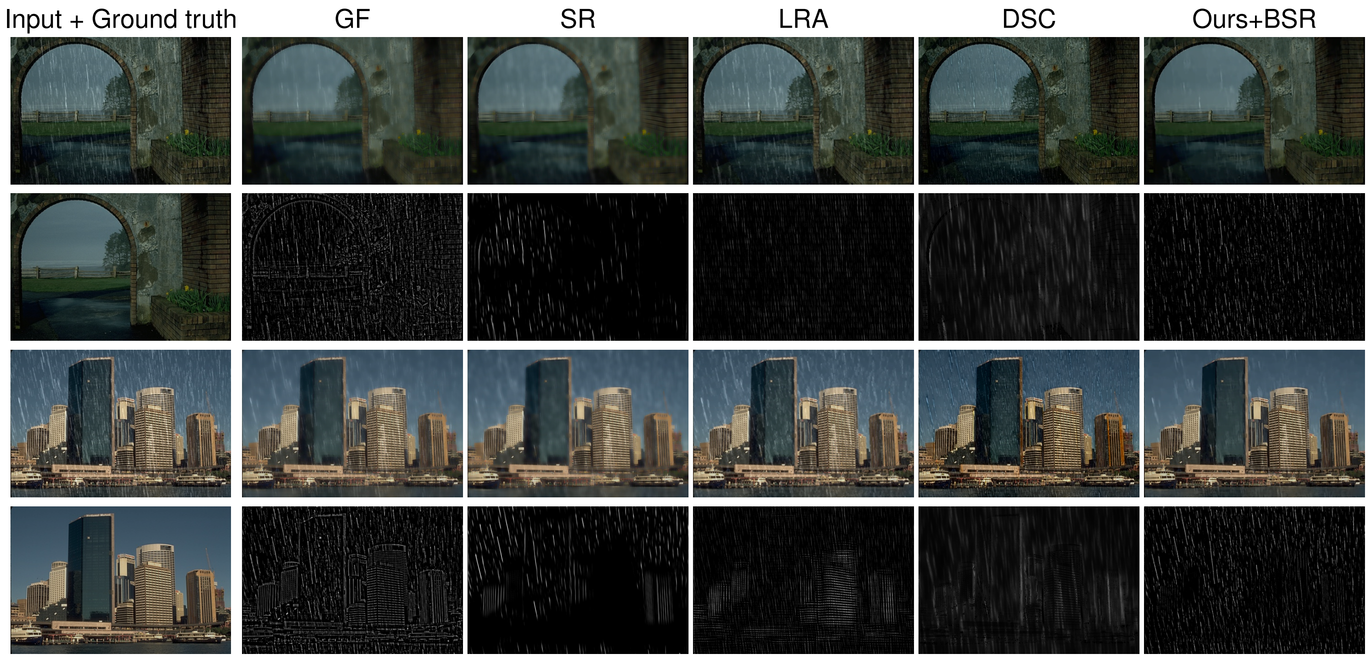


Fig. 8: Visual comparison of different rain streak removal methods on a synthetic data set.

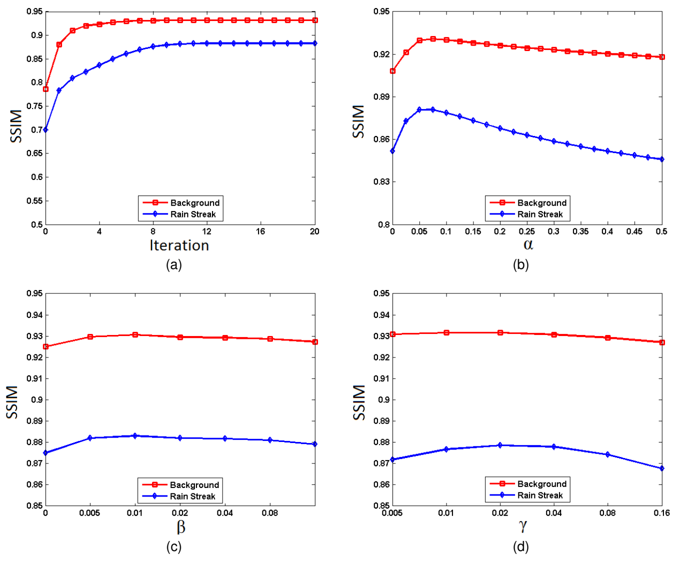


Fig. 7: Study on the parameter settings.

than that of SR [1] (0.7437 / 27.69 dB). An inspection of the recovered image in Figure 8 shows the drawback of SR [1] that it tends to over-smooth the background layer. LRA [2] fails to remove the rain streaks in these two examples in Figure 8. While DSC [8], with a higher overall SSIM and PSNR (0.8657 / 30.96 dB), can obtain slightly better visual results, it still leaves many thin rain structures in the background images. Our original method obtains SSIM = 0.9143 and PSNR = 32.85 dB overall, which is far better than SR [1], LRA [2], and DSC [8]. Our method with the background-structure-residue recovery step shows further improvement over our original method, and obtains the best background recovery on most cases, and also obtains the best overall performance (0.9228 / 33.91 dB). Visually, our full method with background-structure-residue procedure (Ours+BSR) obtains much cleaner background recovery than other methods.

For the rain streak layer recovery, GF gets the lowest numerical scores (0.6977 / 28.87 dB). This point also reflects in Figure 8 that GF [40] falsely assigns many image details into the rain streak layer. In contrast, the rest of the rain streak removal methods [1], [2], [8] capture the rain streaks better than GF [40]. It is observable that LRA [2] obtains less accurate rain streak recovery (0.7089 / 29.86 dB) among all rain streak removal methods, indicating that low rank may not be a sufficient prior for this task. The visual example reveals the limitation of using the low rank prior to model the rain streak layer [2] because it may treat other repetitive patterns in the image as rain streaks (e.g., the bricks, the building facades). SR [1] (0.7454 / 30.58 dB) can capture some rain streaks in the scene but cannot distinguish rain streaks in highly textured regions. Ours (0.8462 / 32.10 dB) and Ours+BSR (0.8483 / 32.58 dB) show noticeable advantages over existing methods both numerically and visually.

B. *Results on Real Images*

Figure 9 and Figure 10 show the results and comparisons using real images where the images in Figure 9 are from [8] and the images in Figure 10 are found on the Internet.

Similar to the previous experiments, the drawbacks of SR [1], LRA [2], and DSC [8] are still present. SR [1] always produces over-smooths backgrounds. LRA [2] either over-smooth the background (case 1 in Figure 9 and case 3 in Figure 10) or fails to remove the rain streaks as the low rank assumption is not always met. DSC [8] exhibits good results when the rain streaks are sparse (Figure 9) but the rain streak removal performance will drop in the cases with more

TABLE I: Quantitative comparison of rain streak removal results on our synthetic data sets (12 images).

		Method	#1	#2	#3	#4	#5	#6	#7	#8	#9	#10	#11	#12	Average
Background	SSIM	GF[40]	0.7643	0.8335	0.8792	0.7793	0.7815	0.8552	0.8929	0.8165	0.8126	0.8156	0.7553	0.8287	0.8179
		SR[1]	0.7309	0.7859	0.8349	0.7617	0.6229	0.7347	0.8169	0.7661	0.7331	0.7410	0.6326	0.7637	0.7437
		LRA[2]	0.8277	0.8686	0.7910	0.8478	0.8797	0.8993	0.9245	0.8182	0.8734	0.8252	0.8514	0.8104	0.8514
		DSC[8]	0.8245	0.8816	0.7550	0.9534	0.9150	0.9340	0.9439	0.8108	0.8974	0.8218	0.8487	0.8027	0.8657
		Ours[9]	0.8844	0.9288	**0.9279**	0.9327	0.8956	0.9525	0.9574	0.8970	0.9122	0.8989	0.8638	**0.9205**	0.9143
		Ours+BSR	**0.8911**	**0.9380**	0.8759	**0.9676**	**0.9319**	**0.9693**	**0.9716**	**0.9002**	**0.9396**	**0.9023**	**0.8846**	0.9015	**0.9228**
	PSNR	GF[40]	30.33	30.49	28.03	30.77	27.23	29.42	31.03	29.02	29.41	29.47	27.99	30.00	29.43
		SR[1]	29.89	28.99	26.42	29.94	25.11	25.68	27.52	27.94	27.85	28.09	26.42	28.37	27.69
		LRA[2]	31.04	31.88	25.79	32.46	28.56	30.85	33.38	28.96	30.87	29.71	29.33	29.80	30.22
		DSC[8]	31.31	31.37	26.53	36.00	29.50	31.67	34.83	28.90	32.66	29.46	29.11	30.17	30.96
		Ours[9]	**33.44**	34.65	**30.47**	36.90	29.86	33.48	35.11	31.72	33.16	32.12	30.41	32.84	32.85
		Ours+BSR	33.40	**35.30**	29.90	**39.50**	**30.42**	**37.13**	**39.32**	**31.94**	**34.25**	**32.26**	**30.71**	**32.85**	**33.91**
Rain streak	SSIM	GF[40]	0.7499	0.7358	0.6980	0.6939	0.5499	0.6455	0.7561	0.7268	0.6891	0.7258	0.6423	0.7591	0.6977
		SR[1]	0.7207	0.7750	0.6007	0.8769	0.7294	0.8361	0.9012	0.7552	0.8090	0.7015	0.5548	0.6846	0.7454
		LRA[2]	0.7135	0.7509	0.5931	0.8059	0.7122	0.8247	0.8674	0.6997	0.7528	0.6341	0.5599	0.5925	0.7089
		DSC[8]	0.7020	0.7587	0.5194	0.8836	0.6917	0.8072	0.8809	0.7035	0.7796	0.6750	0.5784	0.6288	0.7175
		Ours[9]	**0.8506**	**0.8802**	0.7679	0.9267	0.7753	0.8866	0.9264	**0.8612**	**0.8661**	**0.8340**	0.7385	0.8405	0.8462
		Ours+BSR	0.8485	0.8793	**0.7920**	**0.9268**	**0.7757**	**0.8872**	**0.9266**	0.8597	0.8659	0.8327	**0.7387**	**0.8462**	**0.8483**
	PSNR	GF[40]	29.65	30.00	27.01	30.64	26.65	29.21	30.85	28.13	28.93	28.75	27.35	29.25	28.87
		SR[1]	30.14	31.89	26.74	33.72	28.61	33.51	36.04	28.92	31.40	29.21	27.47	29.32	30.58
		LRA[2]	29.45	31.00	26.03	33.76	28.16	33.52	35.17	27.44	30.08	28.19	27.45	28.06	29.86
		DSC[8]	29.98	30.05	25.05	34.52	27.91	30.10	33.31	27.59	31.33	28.03	27.75	28.87	29.54
		Ours[9]	**32.35**	34.00	28.74	37.78	29.10	33.30	34.93	30.25	32.69	**31.18**	**29.41**	31.51	32.10
		Ours+BSR	32.14	**34.11**	**28.80**	**38.33**	**29.17**	**35.69**	**37.75**	**30.31**	**32.70**	31.01	29.35	**31.61**	**32.58**

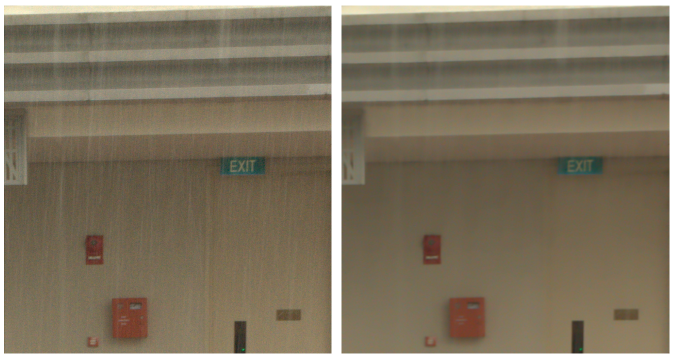


Fig. 11: Rain streak removal on raw data using our method.

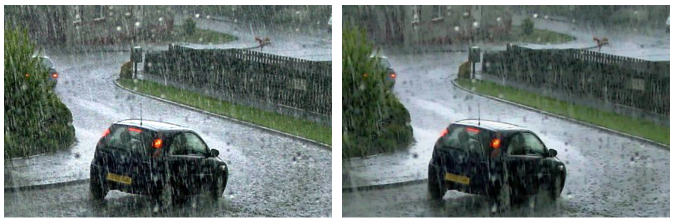


Fig. 12: Image recognition results on the images before and after rain streak removal.

rain streaks (Figure 9). Our method provides more favorable results by effectively removing the rain streaks and meanwhile retaining image details.

Results on raw data In the previous results, we do not assume anything on the camera response function, which is the common practice in the field of rain removal. According to our rain model in Eq.(1), as long as the input image is the linear combination of rain layer and background layer, our proposed method should work properly. However, one might argue that Eq.(1) assumes linearity of the camera response function. To address this concern, we also tested our method on raw image that has linear camera response function. Figure 11 shows that our method works effectively under such a response function.

Application in computer vision Besides visually pleasing output, our method is also helpful in computer vision tasks, such as image recognition. Figure 12 (left) shows an input that is supposed to be a car on the road, but with rain streaks presented. This is a typical example of images taken outside on a rainy day. We use Clarifai,[1] an advanced image recognition system based on a deep convolutional network. This system classifies the input image as $rain$. After the rain streak removal using our method, the output in Figure 12 (right) is correctly labeled as $vehicle$.

V. Conclusion

We have introduced a novel approach for solving the decomposition problem of a background scene and rain streaks using a single image. Unlike existing methods, we impose constraints on both the background and rain layers. These constraints are simple Gaussian mixture models (GMMs) learned from image patches. Based on our experiments, we

[1]https://www.clarifai.com/.

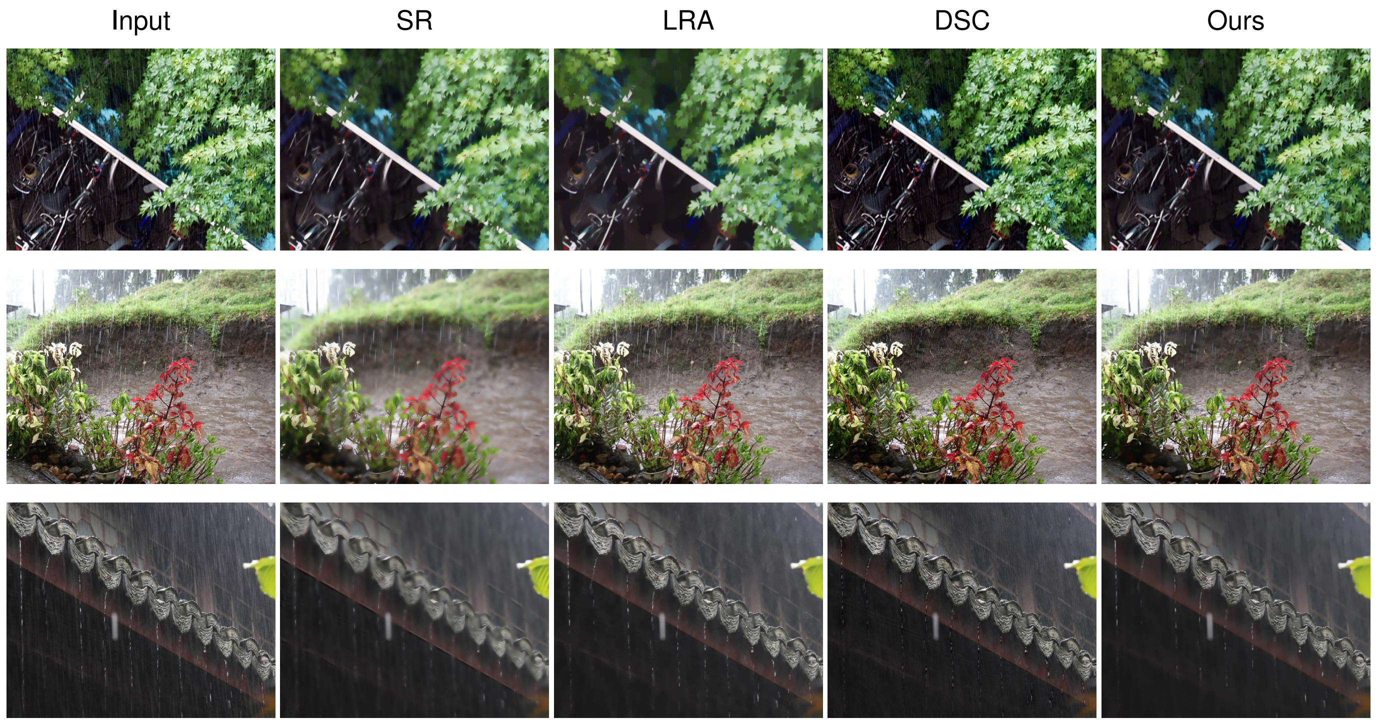


Fig. 9: Visual comparisons of different rain streak removal methods on real images in [8].

showed that these two constraints prove to be more effective than methods based on sparse dictionary learning and low rank constraints.

Our constraint on the rain layer is particularly interesting as rain streaks have special appearances and structures. GMM can effectively capture a considerably narrower distribution to describe rain streaks and distinguish them from the wider range of texture for the background layer. We have verified that this GMM prior for rain streaks is a critical part of the decomposition. Without the GMM priors, the estimated background is much more blurred, and the rain layer contains too much high-frequency textures from the background layer. Our proposed method not only is simple and effective but also does not assume the rain streak orientations, sizes, or scales. It clearly demonstrates the usefulness of the GMM priors to the decomposition framework, which is a step forward in addressing rain streak removal.

In addition, we have proposed a background-structure-residue recovery step to retrieve the incorrectly assigned background details in the rain layers. This step can make the rain streak layer cleaner and also further improve the separation quality. Experimental results have shown that our method quantitatively and qualitatively outperforms existing rain streak removal methods in terms of both background recovery and rain streak recovery.

References

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

[2] Y.-L. Chen and C.-T. Hsu, "A generalized low-rank appearance model for spatio-temporally correlated rain streaks," in *IEEE Int'l Conf. Computer Vision*, 2013.

[3] D. Zoran and Y. Weiss, "From learning models of natural image patches to whole image restoration," in *IEEE Int'l Conf. Computer Vision*, 2011.

[4] S. Roth and M. J. Black, "Fields of experts: A framework for learning image priors," in *IEEE Conf. Computer Vision and Pattern Recognition*, 2005.

[5] Y. Weiss and W. T. Freeman, "What makes a good model of natural images?" in *IEEE Conf. Computer Vision and Pattern Recognition*, 2007.

[6] Y. Shih, D. Krishnan, F. Durand, and W. T. Freeman, "Reflection removal using ghosting cues," in *IEEE Conf. Computer Vision and Pattern Recognition*, 2015.

[7] Y. Li and M. S. Brown, "Exploiting reflection change for automatic reflection removal," in *IEEE Int'l Conf. Computer Vision*, 2013.

[8] Y. Luo, Y. Xu, and H. Ji, "Removing rain from a single image via discriminative sparse coding," in *IEEE Int'l Conf. Computer Vision*, 2015.

[9] Y. Li, R. T. Tan, X. Guo, J. Lu, and M. S. Brown, "Rain streak removal using layer priors," in *IEEE Conf. Computer Vision and Pattern Recognition*, 2016.

[10] R. Fattal, "Single image dehazing," *ACM Trans. Graphics*, vol. 27, no. 3, p. 72, 2008.

[11] R. T. Tan, "Visibility in bad weather from a single image," in *IEEE Conf. Computer Vision and Pattern Recognition*, 2008.

[12] Y. Li, R. T. Tan, and M. S. Brown, "Nighttime haze removal with glow and multiple light colors," in *IEEE Int'l Conf. Computer Vision*, 2015.

[13] K. Garg and S. K. Nayar, "Vision and rain," *Int'l. J. Computer Vision*, vol. 75, no. 1, pp. 3–27, 2007.

[14] ——, "Detection and removal of rain from videos," in *IEEE Conf. Computer Vision and Pattern Recognition*, 2004.

[15] ——, "When does a camera see rain?" in *IEEE Int'l Conf. Computer Vision*, 2005.

[16] X. Zhang, H. Li, Y. Qi, W. K. Leow, and T. K. Ng, "Rain removal in video by combining temporal and chromatic properties," in *IEEE Int'l Conf. Multimedia and Expo*, 2006.

[17] J. Bossu, N. Hautière, and J.-P. Tarel, "Rain or snow detection in image sequences through use of a histogram of orientation of streaks," *Int'l. J. Computer Vision*, vol. 93, no. 3, pp. 348–367, 2011.

[18] P. C. Barnum, S. Narasimhan, and T. Kanade, "Analysis of rain and snow in frequency space," *Int'l. J. Computer Vision*, vol. 86, no. 2-3, pp. 256–274, 2010.

[19] V. Santhaseelan and V. K. Asari, "Utilizing local phase information to

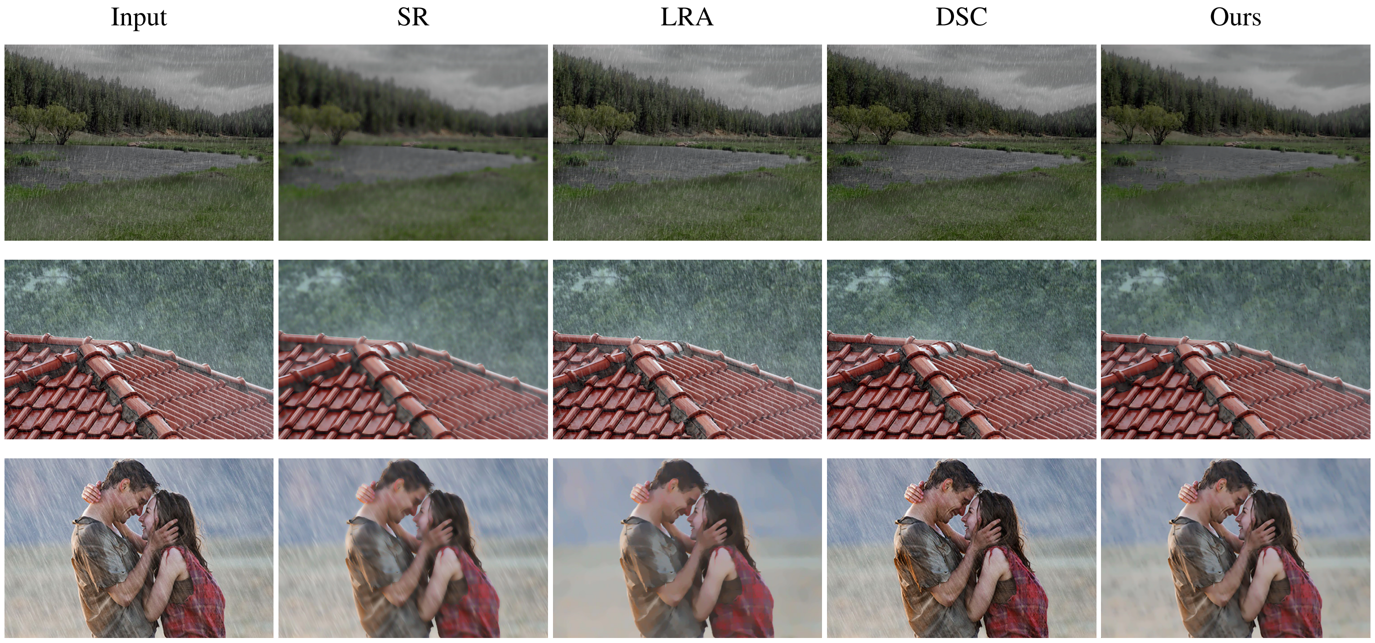


Fig. 10: Visual comparison of different rain streak removal methods on more real example images.

remove rain from video," *Int'l. J. Computer Vision*, vol. 112, no. 1, pp. 71–89, 2015.

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

[21] T. H. Oh, Y. Matsushita, Y. W. Tai, and I. S. Kweon, "Fast randomized singular value thresholding for nuclear norm minimization," in *IEEE Conf. Computer Vision and Pattern Recognition*, 2015.

[22] 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.

[23] D.-A. Huang, L.-W. Kang, Y.-C. F. Wang, and C.-W. Lin, "Self-learning based image decomposition with applications to single image denoising," *IEEE Trans. Multimedia*, vol. 16, no. 1, pp. 83–93, 2014.

[24] S.-H. Sun, S.-P. Fan, and Y.-C. F. Wang, "Exploiting image structural similarity for single image rain removal," in *IEEE Int'l Conf. Image Processing*, 2014, pp. 4482–4486.

[25] J.-H. Kim, C. Lee, J.-Y. Sim, and C.-S. Kim, "Single-image deraining using an adaptive nonlocal means filter," in *IEEE Int'l Conf. Image Processing*, 2013.

[26] D. Eigen, D. Krishnan, and R. Fergus, "Restoring an image taken through a window covered with dirt or rain," in *IEEE Int'l Conf. Computer Vision*, 2013.

[27] H. Kurihata, T. Takahashi, I. Ide, Y. Mekada, H. Murase, Y. Tamatsu, and T. Miyahara, "Rainy weather recognition from in-vehicle camera images for driver assistance," in *IEEE Intelligent Vehicles Symposium*, 2005.

[28] M. Roser, J. Kurz, and A. Geiger, "Realistic modeling of water droplets for monocular adherent raindrop recognition using bezier curves," in *Asian Conf. Computer Vision*, 2011.

[29] S. You, R. T. Tan, R. Kawakami, and K. Ikeuchi, "Adherent raindrop detection and removal in video," in *IEEE Conf. Computer Vision and Pattern Recognition*, 2013.

[30] ——, "Adherent raindrop modeling, detection and removal in video," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. PP, no. 99, p. 1.

[31] Y. Li and M. S. Brown, "Single image layer separation using relative smoothness," in *IEEE Conf. Computer Vision and Pattern Recognition*, 2014.

[32] D. Geman and C. Yang, "Nonlinear image recovery with half-quadratic regularization," *IEEE Trans. Image Processing*, vol. 4, no. 7, pp. 932–946, 1995.

[33] C. Zhu, R. H. Byrd, P. Lu, and J. Nocedal, "Algorithm 778: L-bfgs-b: Fortran subroutines for large-scale bound-constrained optimization," *ACM Trans. on Mathematical Software*, vol. 23, no. 4, pp. 550–560, 1997.

[34] Z. Hu and M.-H. Yang, "Good regions to deblur," in *European Conf. Computer Vision*, 2012.

[35] G. Meng, Y. Wang, J. Duan, S. Xiang, and C. Pan, "Efficient image dehazing with boundary constraint and contextual regularization," in *IEEE Int'l Conf. Computer Vision*, 2013.

[36] Z. Farbman, R. Fattal, D. Lischinski, and R. Szeliski, "Edge-preserving decompositions for multi-scale tone and detail manipulation," *ACM Trans. Graph.*, vol. 27, no. 3, 2008.

[37] D. Min, S. Choi, J. Lu, B. Ham, K. Sohn, and M. N. Do, "Fast global image smoothing based on weighted least squares," *IEEE Trans. Image Processing*, vol. 23, no. 12, pp. 5638–5653, 2014.

[38] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: from error visibility to structural similarity," *IEEE Trans. Image Processing*, vol. 13, no. 4, pp. 600–612, 2004.

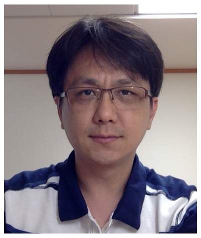
[39] L. I. Rudin, S. Osher, and E. Fatemi, "Nonlinear total variation based noise removal algorithms," *Physica D: Nonlinear Phenomena*, vol. 60, no. 1, pp. 259–268, 1992.

[40] K. He, J. Sun, and X. Tang, "Guided image filtering," in *European Conf. Computer Vision*, 2010.

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

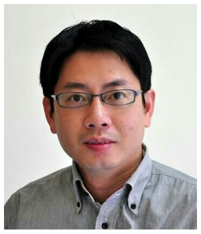
[42] D. Martin, C. Fowlkes, D. Tal, and J. Malik, "A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics," in *IEEE Int'l Conf. Computer Vision*, 2001.

Yu Li received his Ph.D. degree in National University of Singapore in 2015. He is now with Advanced Digital Sciences Center, a research center founded by University of Illinois at Urbana-Champaign (UIUC) and the Agency for Science, Technology and Research (A*STAR), Singapore. His research interests include computer vision, computational photography, and computer graphics.



Robby T. Tan Robby T. Tan received the PhD degree in computer science from the University of Tokyo. He is now an assistant professor at both Yale-NUS College and ECE (Electrical and Computing Engineering), National University of Singapore. Previously, he was an assistant professor at Utrecht University. His research interests include machine learning and computer vision, particularly in dealing with bad weather, physics-based and motion analysis. He is a member of the IEEE.

Xiaojie Guo received the B.E. degree in software engineering from the School of Computer Science and Technology, Wuhan University of Technology, Wuhan, China, in 2008, and the M.S. and Ph.D. degrees in computer science from the School of Computer Science and Technology, Tianjin University, Tianjin, China, in 2010 and 2013, respectively. He is currently an Associate Professor with the Institute of Information Engineering, Chinese Academy of Sciences. He was a recipient of the Piero Zamperoni Best Student Paper Award in the International Conference on Pattern Recognition (International Association on Pattern Recognition), in 2010.



Jiangbo Lu received his B.S. and M.S. degrees in electrical engineering from Zhejiang University, Hangzhou, China, in 2000 and 2003, respectively, and the Ph.D. degree in electrical engineering, Katholieke Universiteit Leuven, Leuven, Belgium, in 2009. From 2009 to 2016, he was with the Advanced Digital Sciences Center (ADSC), Singapore, which is a joint research center between the University of Illinois at Urbana-Champaign (UIUC), USA, and the Agency for Science, Technology and Research (A*STAR), Singapore, where he has led several research projects as a Senior Research Scientist. Since 2017, he has joined Shenzhen Cloudream Technology Co., Ltd. as the Chief Technology Officer (CTO), and is leading R & D departments to work on cutting-edge problems broadly in computer vision, computer graphics, and machine learning centered on perceiving and understanding humans. His research interests include computer vision, visual computing, robotic vision, and computational imaging. Dr. Lu served as an Associate Editor for IEEE Transactions on Circuits and Systems for Video Technology (TCSVT) in 2012-2016. He received the 2012 TCSVT Best Associate Editor Award.

Michael S. Brown obtained his BS and PhD in Computer Science from the University of Kentucky in 1995 and 2001 respectively. He is a professor at York University in the Lassonde School of Engineering. Dr. Browns research interests include computer vision, image processing and computer graphics.