Robust Video Content Alignment and Compensation for Rain Removal in a CNN Framework

Jie Chen[1], Cheen-Hau Tan[1], Junhui Hou[2], Lap-Pui Chau[1], and He Li[3]
[1]School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore
[2]Department of Computer Science, City University of Hong Kong
[3]Singapore Technologies Dynamics Pte Ltd
{chen.jie, cheenhau, elpchau}@ntu.edu.sg, jh.hou@cityu.edu.hk, lihe@stengg.com

Abstract

Rain removal is important for improving the robustness of outdoor vision based systems. Current rain removal methods show limitations either for complex dynamic scenes shot from fast moving cameras, or under torrential rain fall with opaque occlusions. We propose a novel derain algorithm, which applies superpixel (SP) segmentation to decompose the scene into depth consistent units. Alignment of scene contents are done at the SP level, which proves to be robust towards rain occlusion and fast camera motion. Two alignment output tensors, i.e., optimal temporal match tensor and sorted spatial-temporal match tensor, provide informative clues for rain streak location and occluded background contents to generate an intermediate derain output. These tensors will be subsequently prepared as input features for a convolutional neural network to restore high frequency details to the intermediate output for compensation of mis-alignment blur. Extensive evaluations show that up to 5dB reconstruction PSNR advantage is achieved over state-of-the-art methods. Visual inspection shows that much cleaner rain removal is achieved especially for highly dynamic scenes with heavy and opaque rainfall from a fast moving camera.

1. Introduction

Modern intelligent systems rely more and more on visual information as input. However, in an outdoors setting, visual input quality and in turn, system performance, could be seriously degraded by atmospheric turbulences [11, 23]. One such turbulence, rain streaks, degrade image contrast and visibility, obscure scene features, and could be misconstrued as scene motion by computer vision algorithms. Rain removal is therefore vital to ensure the robustness of outdoor vision-based systems.

There are two categories of methods for rain removal – image-based methods, which rely solely on the information of the processed frame, and video-based methods, which also utilize temporal clues from neighboring frames. Due to the lack of temporal information, image-based methods face difficulties in recovering from torrential rain with large and opaque occlusions.

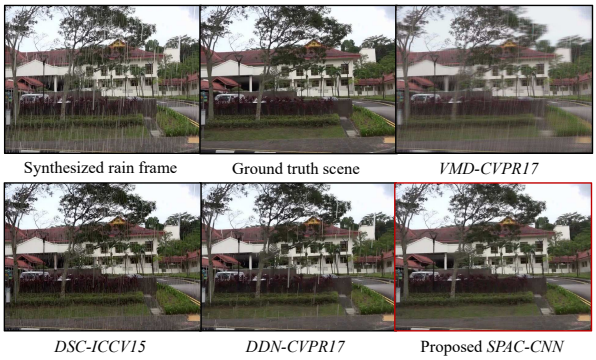


Figure 1: Comparison of derain outputs by different algorithms for a challenging video sequence with fast camera motion and heavy rain fall. Image-based derain methods, i.e., discriminative sparse coding (*DSC*) [22] and deep detail network (*DDN*) [8] fail to remove large and/or opaque rain streaks. A video-based method via matrix decomposition (*VMD*) [24] creates serious blur due to fast camera motion. Our proposed *SPAC-CNN* can cleanly remove the rain streaks and preserve scene contents truthfully.

To properly utilize temporal information, video-based methods require scene content to be aligned throughout consecutive frames. However, this requirement is challenging due to two factors – motion of the camera and dynamic scene content, i.e., presence of moving object. Previous works tackle these two issues separately. Camera motion-induced scene content shifts can be reversed using global frame alignment [29, 27]. However, the granularity of global alignment is too large when scene depth range is large; parts of scene content will be poorly aligned. Scene

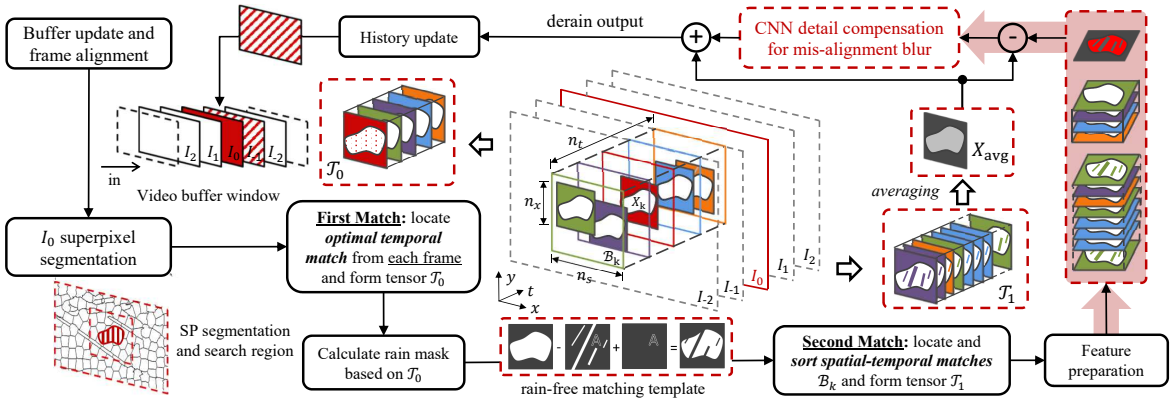


Figure 2: System diagram for the proposed *SPAC-CNN* rain removal algorithm.

content shifts due to scene object motions could cause the scene to be misclassified as rain. One solution is to identify and exclude these pixels. This approach, however, is unable to remove rain that overlaps moving objects.

In this paper, we propose a novel and elegant framework that simultaneously solves both issues for the video-based approach – rain removal based on robust SuperPixel (SP) Alignment between video frames followed by detail Compensation in a CNN framework (*SPAC-CNN*). First, the target video frame is segmented into SPs and each SP is aligned with its temporal neighbors. This step simultaneously aligns both the scene background and moving objects without prior assumptions about moving objects. Scene content is also much better aligned at a SP level granularity. An intermediate derain output can be obtained by averaging the aligned SPs, which unavoidably introduces blurring. We restore the rain free details to the intermediate output by extracting the information from the aligned SPs using a convolutional neural network (CNN).

Extensive experiments show that our proposed algorithm achieves up to 5*dB* reconstruction advantage over state-of-the-art rain removal methods. Visual inspection shows that rain is much better removed, especially for heavy and opaque rainfall regions over highly dynamic scene content. Fig. 1 illustrates the advantage of our proposed algorithm over existing methods in a challenging video sequence. The contribution of this work can be generalized as follows:

1. We propose a novel spatial-temporal content alignment algorithm at SP level, which can handle fast camera motion and dynamic scene contents in one framework. This mechanism greatly outperforms existing scene motion analysis methods that models background and foreground motion separately.
2. The strong local properties of SPs can robustly counter heavy rain interferences, and facilitate much more accurate alignment. Owing to such robust alignment, accurate temporal correspondence could be established for rain occlusions such that heavily occluded backgrounds could be truthfully restored. This greatly outperforms image-based derain methods in which recovery of large and opaque rain occlusions remain the biggest challenge.
3. We propose a set of very efficient spatial-temporal features for the compensation of high frequency details lost during the deraining process. An efficient CNN network is designed, and a synthetic rain video dataset is created for training the CNN.

2. Related Work

Rain removal based on a single image is intrinsically a challenging one, since it only relies on visual features and priors to distinguish rain from the background. Local photometric, geometric, and statistical properties of rain have been studied in [11, 10, 36, 15]. Li et al. [20] models background and rain streaks as layers to be separated. Under the sparse coding framework, rain and backgrounds can be efficiently separated either with classified dictionary atoms [13, 6], or via discriminative sparse coding [22]. Convolutional Neural Networks have been very effective in both high-level vision tasks [19] and low-level vision applications for capturing signal characteristics [14, 34]. Hence, different network structures and features were explored for rain removal, such as the deep detail network [8], and the joint rain detection and removal model [32]. Due to the lack of temporal information, heavy and opaque rain is difficult to be distinguished from scene structures. Full recovery of a seriously occluded scene is almost impossible.

The temporal information from a video sequence provides huge advantage for rain removal [9, 3, 25, 26, 33]. True rain pixels are separated from moving object pixels based on statistics of intensity values [29] or chromatic values [35], on geometric properties of connected candidate pixels [5], or on segmented motion regions [7]. Kim's work [16] compensates for scene content motion by using optical flow for content alignment. Ren et al. [24] decomposes a video into background, rain, and moving objects using matrix decomposition. Moving objects are derained by tempo-

rally aligning them using patch matching, while the moving camera effect is modeled using a frame transform variable. Temporal derain methods can handle occlusions much better than image-based methods; however, these methods perform poorly for complex dynamic scenes shot from fast moving cameras.

3. Proposed Model

Throughout the paper, scalars are denoted by italic lower-case letters, 2D matrices by upper-case letters, 3D tensors, functions, and operators by script letters.

Given a target derain video frame I_0, we look at its immediate past and future neighbor frames to create a sliding buffer window of length n_t: $\{I_i | i = [-\frac{n_t-1}{2}, \frac{n_t-1}{2}]\}$. Here, negative and positive i indicate past and future frames, respectively. We only derain the Y luminance channel. The derain output is used to update the history buffer (Fig. 2). Such history update mechanism ensures cleaner derain for heavy rainfall scenarios.

The system diagram for the proposed *SPAC-CNN* rain removal algorithm is shown in Fig. 2. The algorithm can be divided into two parts: first, video content alignment is carried out at SP level, which consists two SP template matching operations that produce two output tensors: the ***optimal temporal match tensor*** $\mathcal{T}_0$, and the ***sorted spatial-temporal match tensor*** $\mathcal{T}_1$. An intermediate derain output X_{avg} is calculated by averaging the *slices*[1] of the tensor $\mathcal{T}_1$. Second, these two tensors will be prepared as input features to a CNN to compensate the high frequency details lost in X_{avg} caused by mis-alignment blur. The detail of each component will be explained in this section.

3.1. Robust Content Alignment via Superpixel Spatial-Temporal Matching

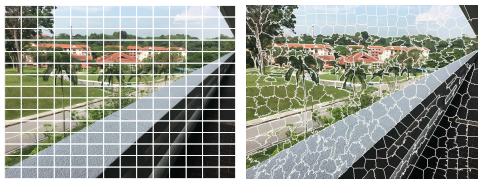
One of the most important procedure for video-based derain algorithms is the estimation of content correspondence between video frames. With accurate content alignment, rain occlusions could be easily detected and removed with information from the temporal axis.

3.1.1 Content Alignment: Global vs. Superpixel

The popular solution to compensate camera motion between two frames is via a homography transform matrix estimated based on global consensus of a group of matched feature points [4, 28]. Due to the reasons analyzed in Sec. 1, perfect content alignment can never be achieved for all pixels with a global transform at whole frame level, especially for dynamic scenes with large depth range.

The solution naturally turns to pixel-level alignment, which faces no fewer challenges: first, feature points are sparse, and feature-less regions are difficult to align; more importantly, rain streak occlusions will cause serious interferences to feature matching at single pixel level. Information from larger areas are required to overcome rain interferences. This lead us to our final solution: to decompose images into smaller depth consistent units.

[1]A slice is a two-dimensional section of a higher dimensional tensor, defined by fixing all but two indices [18].



(a) (b)

Figure 3: Rectangular and SP segmentation units.

The concept of SuperPixel (SP) is to group pixels into perceptually meaningful atomic regions [2, 30, 21]. Boundaries of SP usually coincide with those of the scene contents. Comparing Fig. 3(a) and (b), the SPs are very adaptive in shape, and are more likely to segment uniform depth regions compared with rectangular units. We adopt SP as the basic unit for content alignment.

3.1.2 Optimal Temporal Matching for Rain Detection

Let $\mathcal{P}_k$ denote the set of pixels that belong to the k-th SP on I_0. Let $X_k \in \mathbb{R}^{n_x \times n_x}$ be the bounding box that covers all pixels in $\mathcal{P}_k$ ($\mathcal{P}_k \subset X_k$). Let $\mathcal{B}_k \in \mathbb{R}^{n_s \times n_s \times n_t}$ denote a spatial-temporal buffer centered on $\mathcal{P}_k$. As illustrated in Fig. 2, $\mathcal{B}_k$ spans the entire sliding buffer window, and its spatial range $n_s \times n_s$ is set to cover the possible motion range of $\mathcal{P}_k$ in its neighboring frames.

Pixels within the same SP are very likely to belong to the same object and possess identical motion between adjacent frames. Therefore, we can approximate the SP appearance in their adjacent frames based on its appearance in the current frame via linear translations.

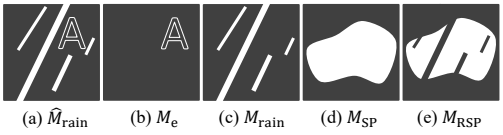
Searching for the reference SP is done by template matching of the target SP at all candidate locations in $\mathcal{B}_k$. A match location is found at frame $I_{t'}$ according to:

$$(\hat{u}, \hat{v}) = \arg\min_{u,v} \sum_{(x,y)\in X_k} |\mathcal{B}_k(x+u, y+v, t') - X_k(x,y)|^2 \odot M_{\text{SP}}(x,y). \quad (1)$$

As shown in Fig. 4(d), M_{SP} indicates SP pixels $\mathcal{P}_k$ in the bounding box X_k. $\odot$ denotes element-wise multiplication. Each match at a different frame becomes a *temporal slice* for the ***optimal temporal match tensor*** $\mathcal{T}_0 \in \mathbb{R}^{n_x \times n_x \times n_t}$:

$$\mathcal{T}_0(\cdot, \cdot, t') = \mathcal{B}_k(x+\hat{u}, y+\hat{v}, t'),\ (x,y) \in X_k. \quad (2)$$

Based on the temporal clues provided by $\mathcal{T}_0$, a rain mask can be estimated. Since rain increases the intensity of its covered pixels [9], rain pixels in X_k are expected to have higher intensity than their collocated temporal neighbors in



(a) $\hat{M}_{\text{rain}}$ (b) M_{e} (c) M_{rain} (d) M_{SP} (e) M_{RSP}

Figure 4: Illustration of various masks and matching templates used in the proposed algorithm.

$\mathcal{T}_0$. We first compute a binary tensor $\mathcal{M}_0 \in \mathbb{R}^{n_x \times n_x \times n_t}$ to detect positive temporal fluctuations:

$$\mathcal{M}_0 = \begin{cases} 1 & \mathcal{R}(X_k, n_t) - \mathcal{T}_0 \geq \epsilon_{\text{rain}} \\ 0 & \text{otherwise} \end{cases}, \tag{3}$$

where operator $\mathcal{R}(\Phi, \psi)$ is defined as replicating the 2D slices $\Phi \in \mathbb{R}^{n_1 \times n_2}$ ψ times and stacking along the thrid dimension into a tensor of $\mathbb{R}^{n_1 \times n_2 \times \psi}$. To robustly handle re-occurring rain streaks, we classify pixels as rain when at least 3 positive fluctuations are detected in $\mathcal{M}_0$. An initial rain mask $\hat{M}_{\text{rain}} \in \mathbb{R}^{n_x \times n_x}$ can be calculated as:

$$\hat{M}_{\text{rain}}(x, y) = [\sum_t \mathcal{M}_0(x, y, t)] \geq 3. \tag{4}$$

Due to possible mis-alignment, edges of background could be misclassified as rain. Since rain steaks don't affect values in the chroma channels (Cb and Cr), a rain-free edge map M_{e} could be calculated by thresholding the sum of gradients of these two channels with ϵ_{e}. The final rain mask $M_{\text{rain}} \in \mathbb{R}^{n_s \times n_s}$ is calculated as:

$$M_{\text{rain}} = \hat{M}_{\text{rain}} \odot (1 - M_e). \tag{5}$$

A visual demonstration of $\hat{M}_{\text{rain}}$, M_{e}, and M_{rain} is shown in Fig. 4(a), (b), and (c), respectively. In our implementation, ϵ_{rain} is set to 0.012 while ϵ_{edge} is set to 0.2.

3.1.3 Sorted Spatial-Temporal Template Matching for Rain Occlusion Suppression

The second round of template matching will be carried out based on the following cost function:

$$E(u, v, t) = \sum_{(x,y) \in X_k} |\mathcal{B}_k(x + u, y + v, t) - X_k(x, y)|^2 \odot M_{\text{RSP}}(x, y). \tag{6}$$

The **rain-free matching template** M_{RSP} is calculated as:

$$M_{\text{RSP}} = M_{\text{SP}} \odot (1 - M_{\text{rain}}). \tag{7}$$

As shown in Fig. 4(e), only the ***rain-free background SP pixels*** will be used for matching. Each candidate locations in $\mathcal{B}_k$ (except current frame $\mathcal{B}_k(\cdot, \cdot, 0)$) are sorted in ascending order based on their cost E defined in Eq. (6). The top n_{st} candidates with smallest E will be stacked as slices to form the ***sorted spatial-temporal match tensor*** $\mathcal{T}_1 \in \mathbb{R}^{n_s \times n_s \times n_{st}}$.

The slices of $\mathcal{T}_1(\cdot, \cdot, t)$ are expected to be well-aligned to the current target SP $\mathcal{P}_k$, and is robust to interferences from the rain. Since rain pixels are temporally randomly and sparsely distributed within $\mathcal{T}_1$, when n_{st} is sufficiently large, we can get a good estimation of the rain free image through *tensor slice averaging*, which functions to suppress rain induced intensity fluctuations, and bring out the occluded background pixels:

$$X_{\text{avg}} = \frac{\sum_t \mathcal{T}_1(\cdot, \cdot, t)}{n_{st}}. \tag{8}$$

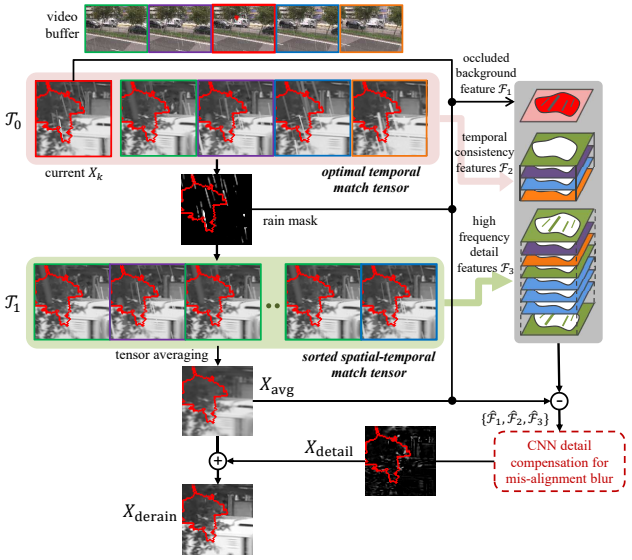


Figure 5: Illustration of feature preparation for the detail recovery CNN.

Fig. 5 gives a visual example of X_{avg} and its calculation flow. We can see that all rain streaks have been suppressed in X_{avg} after the averaging.

3.2. Detail Compensation for Mis-Alignment Blur

The averaging of $\mathcal{T}_1$ slices provides a good estimation of rain free image; however, it creates noticeable blur due to un-avoidable mis-alignment, especially when the camera motion is fast. To compensate the lost high frequency content details without reintroducing the rain streaks, we propose to use a CNN model for the task.

3.2.1 Occluded Background Feature

X_{avg} from Eq. (8) can be used as one important clue to recover rain occluded pixels. Rain streak pixels indicated by the rain mask M_{rain} are replaced with corresponding pixels from X_{avg} to form the first feature $\mathcal{F}_1 \in \mathbb{R}^{n_x \times n_x \times 1}$:

$$\mathcal{F}_1 = X_k \odot (1 - M_{\text{rain}}) + X_{\text{avg}} \odot M_{\text{rain}}. \tag{9}$$

Note that the feature $\mathcal{F}_1$ itself is already a reasonable derain output. However its quality is greatly limited by the correctness of the rain mask M_{rain}. For **false positive**[2] rain

[2]False positive rain pixels refer to background pixels falsely classified as rain; false negative rain pixels refer to rain pixels falsely classified as background.

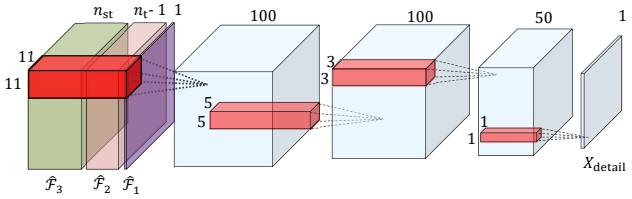


Figure 6: CNN architecture for compensation of mis-alignment blur. Each convolutional layer is followed by a rectified linear unit (ReLU).

pixels, X_{avg} will introduce content detail loss; for **false negative** pixels, rain streaks will be added back from X_k. This calls for more informative features.

3.2.2 Temporal Consistency Feature

The temporal consistency feature is designed to handle **false negative** rain pixels in M_{rain}, which falsely add rain streaks back to $\mathcal{F}_1$. For a correctly classified and recovered pixel (a.k.a. **true positive**) in Eq. (9), intensity consistency should hold such that for the collocated pixels in the neighboring frames, there are only positive intensity fluctuations caused by rain in *those* frames. Any obvious negative intensity drop along the temporal axis is a strong indication that such pixel is a **false negative** pixel.

The temporal slices in $\mathcal{T}_0$ establishes optimal temporal correspondence at each frame, which embeds enough information for the CNN to deduce the above analyzed false negative logic, therefore they shall serve as the second feature $\mathcal{F}_2 \in \mathbb{R}^{n_x \times n_x \times (n_t-1)}$:

$$\mathcal{F}_2 = \{\mathcal{T}_0(\cdot,\cdot,t) |\ t = [-\frac{n_t-1}{2}, \frac{n_t-1}{2}],\ t \neq 0\}. \quad (10)$$

3.2.3 High Frequency Detail Feature

The matched slices in $\mathcal{T}_1$ are sorted according to their rain-free resemblance to X_k, which provide good reference to the content details with supposedly small mis-alignment. We directly use the tensor $\mathcal{T}_1$ as the last group of features $\mathcal{F}_3 = \mathcal{T}_1 \in \mathbb{R}^{n_x \times n_x \times n_{st}}$. This feature will compensate the detail loss introduced by the operations in Eq. (9) for **false positive** rain pixels.

In order to facilitate the network training, we limit the mapping range between the input features and regression output by removing the low frequency component (X_{avg}) from these input features. Pixels in X_k but outside of the SP $\mathcal{P}_k$ is masked out with M_{SP}:

$$\begin{aligned} \hat{\mathcal{F}}_1 &= (\mathcal{F}_1 - X_{\text{avg}}) \odot M_{\text{SP}}, \quad (11)\\ \hat{\mathcal{F}}_2 &= (\mathcal{F}_2 - \mathcal{R}(X_{\text{avg}}, n_t-1)) \odot \mathcal{R}(M_{\text{SP}}, n_t-1),\\ \hat{\mathcal{F}}_3 &= (\mathcal{F}_3 - \mathcal{R}(X_{\text{avg}}, n_{st})) \odot \mathcal{R}(M_{\text{SP}}, n_{st}). \end{aligned}$$

The final input feature set is $\{\hat{\mathcal{F}}_1,\ \hat{\mathcal{F}}_2,\ \hat{\mathcal{F}}_3\}$. The feature preparation process is summarized in Fig. 5.

Figure 7: 8 testing rainy scenes synthesized with *Adobe After Effects* [1]. First row (Group *a*) are shot with a panning unstable camera. Second row (Group *b*) are from a fast moving camera (speed range between 20 to 30 *km/h*)

3.2.4 CNN Structure and Training Details

The CNN architecture is designed as shown in Fig. 6. The network consists of four convolutional layers with decreasing kernel sizes of 11, 5, 3, and 1. All layers are followed by a rectified linear unit (ReLU). Our experiments show this fully convolutional network is capable of extracting useful information from the input features and efficiently providing reliable predictions of the content detail $X_{\text{detail}} \in \mathbb{R}^{n_x \times n_x \times 1}$. The final rain removal output will be:

$$X_{\text{derain}} = X_{\text{avg}} + X_{\text{detail}}. \quad (12)$$

For the CNN training, we minimize the $\mathcal{L}_2$ distance between the derain output and the ground truth scene:

$$E = [\hat{X} - X_{\text{avg}} - X_{\text{detail}}]^2, \quad (13)$$

here $\hat{X}$ denotes the ground truth clean image. We use stochastic gradient descent (SGD) to minimize the objective function. Mini-batch size is set as 50 for better trade-off between speed and convergence. The Xavier approach [12] is used for network initialization, and the ADAM solver [17] is adpatoed for system training, with parameter settings $\beta_1 = 0.9$, $\beta_2 = 0.999$, and learning rate $\alpha = 0.0001$.

To create the training rain dataset, we first took a set of 8 rain-free VGA resolution video clips of various city and natural scenes. The camera was of diverse motion for each clip, e.g., panning slowly with unstable movements, or mounted on a fast moving vehicle with speed up to 30 *km/h*. Next, rain was synthesized over these video clips with the commercial editing software *Adobe After Effects* [1], which can create realistic synthetic rain effect for videos with adjustable parameters such as raindrop size, opacity, scene depth, wind direction, and camera shutter speed. This provides us a diverse rain visual appearances for the network training.

We synthesized 3 to 4 different rain appearances with different synthetic parameters over each video clip, which provides us 25 rainy scenes. For each scene, 21 frames were randomly extracted (together with their immediate buffer window for calculating features). Each scene was segmented into approximately 300 SPs, therefore finally we have around 157,500 patches in the training dataset.

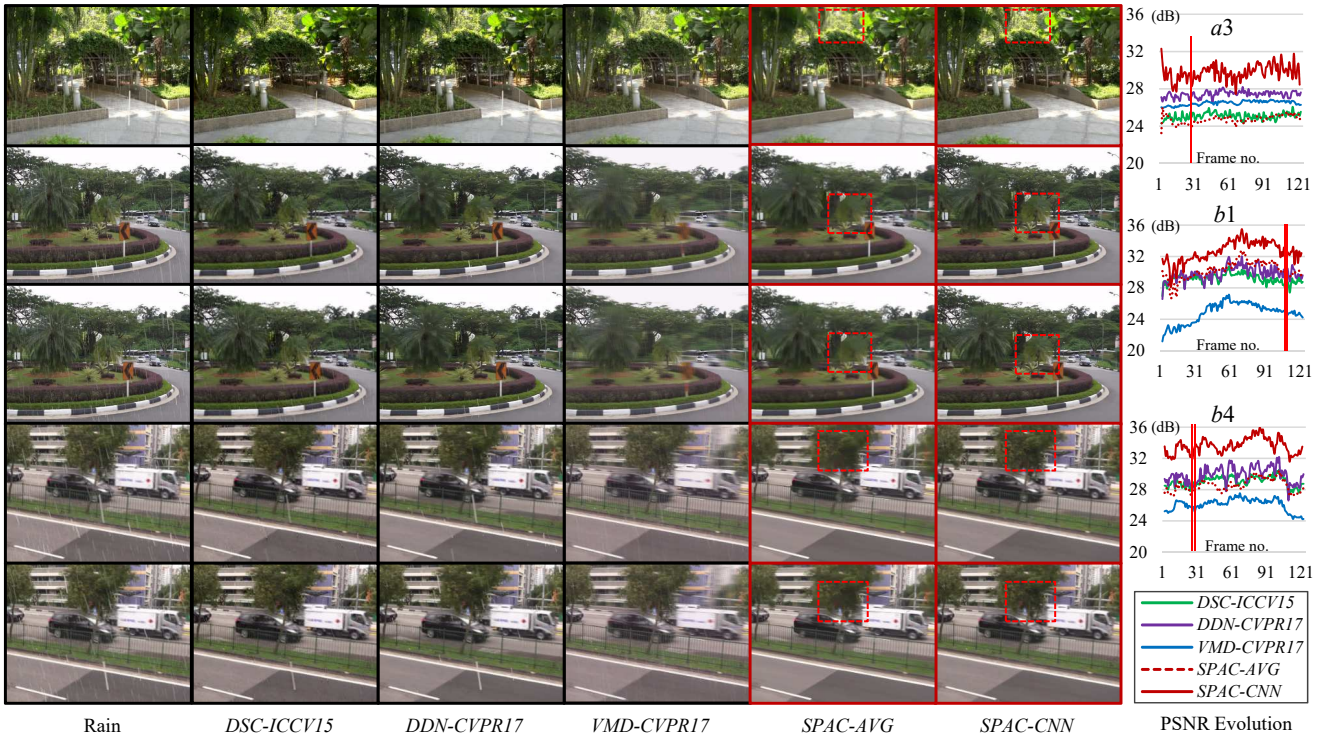


Figure 8: Visual comparison for different rain removal methods on the synthetic testing data *a*3 (row 1), two consecutive frames for the data *b*1 (row 2-3), and *b*4 (row 4-5). The PSNR evolution curves for each frame are shown on the right.

Table 1: Rain removal performance comparison between different methods in terms of scene reconstruction PSNR/SSIM, and F-measure for rain streak edge PR curves.

Camera Motion	*Clip No.*	Rain		*DSC-ICCV15* [22]			*DDN-CVPR17* [8]			*VMD-CVPR17* [24]			*SPAC-Avg*			*SPAC-CNN*		
		PSNR	SSIM	F	PSNR	SSIM	F	PSNR	SSIM	F	PSNR	SSIM	F	PSNR	SSIM	F	PSNR	SSIM
panning unstable camera	*a*1	28.46	0.94	0.38	25.61	0.93	0.47	28.02	0.95	0.47	26.96	0.92	0.39	24.78	0.87	**0.51**	**29.78**	**0.97**
	*a*2	28.09	0.95	0.33	27.11	0.95	0.44	27.38	0.95	**0.51**	24.80	0.93	0.40	26.34	0.89	**0.51**	**30.09**	**0.96**
	*a*3	27.84	0.93	0.43	25.08	0.92	0.45	27.41	0.94	0.42	26.45	0.90	0.40	24.72	0.85	**0.54**	**29.75**	**0.96**
	*a*4	31.48	0.95	0.34	28.82	0.95	0.53	32.47	0.97	**0.55**	29.55	0.94	0.48	29.90	0.93	0.54	**34.82**	**0.98**
	avg. *a*	28.97	0.94	0.37	26.66	0.94	0.47	28.82	0.95	0.49	26.94	0.92	0.42	26.44	0.89	**0.53**	**31.11**	**0.97**
camera speed 20-30 km/h	*b*1	28.72	0.92	0.42	28.78	0.92	0.53	29.48	0.96	0.35	24.09	0.84	0.47	26.35	0.89	**0.55**	**31.19**	**0.96**
	*b*2	29.49	0.90	0.43	29.58	0.92	0.50	30.23	0.95	0.43	25.81	0.89	0.50	28.83	0.93	**0.57**	**34.05**	**0.98**
	*b*3	31.04	0.95	0.33	29.55	0.95	**0.53**	31.39	0.97	0.43	26.12	0.90	0.48	29.55	0.94	**0.53**	**33.73**	**0.98**
	*b*4	27.99	0.92	0.50	29.10	0.93	0.51	29.83	0.96	0.48	25.90	0.88	0.53	28.85	0.92	**0.58**	**33.79**	**0.97**
	avg. *b*	29.31	0.92	0.42	29.25	0.93	0.52	30.23	0.96	0.42	25.48	0.88	0.50	28.40	0.92	**0.56**	**33.19**	**0.97**

4. Performance Evaluation

We set the sliding video buffer window size $n_t = 5$. Each VGA resolution frame was segmented into around 300 SPs using the SLIC method [2]. The bounding box size was $n_x = 80$, and the spatial-temporal buffer $\mathcal{B}_k$ dimension was $n_s \times n_s \times n_t = 30 \times 30 \times 5$. MatConvNet [31] was adopted for model training, which took approximately 54 hours to converge over the training dataset introduced in Sec. 3.2.4. The training and all subsequent experiments were carried out on a desktop with Intel E5-2650 CPU, 56GB RAM, and NVIDIA GeForce GTX 1070 GPU.

4.1. Quantitative Evaluation

To quantitatively evaluate our proposed algorithm, we took a set of 8 videos (different from the training set), and synthesized rain over these videos with varying parameters. Each video is around 200 to 300 frames. All subsequent results shown for each video are the average of all frames.

To test the algorithm performance in handling cameras with different motion, we divided the 8 testing scenes into two groups: Group *a* consists of scenes shot from a panning and unstable camera; Group *b* from a fast moving camera (with speed range between 20 to 30 *km/h*). Thumbnails and

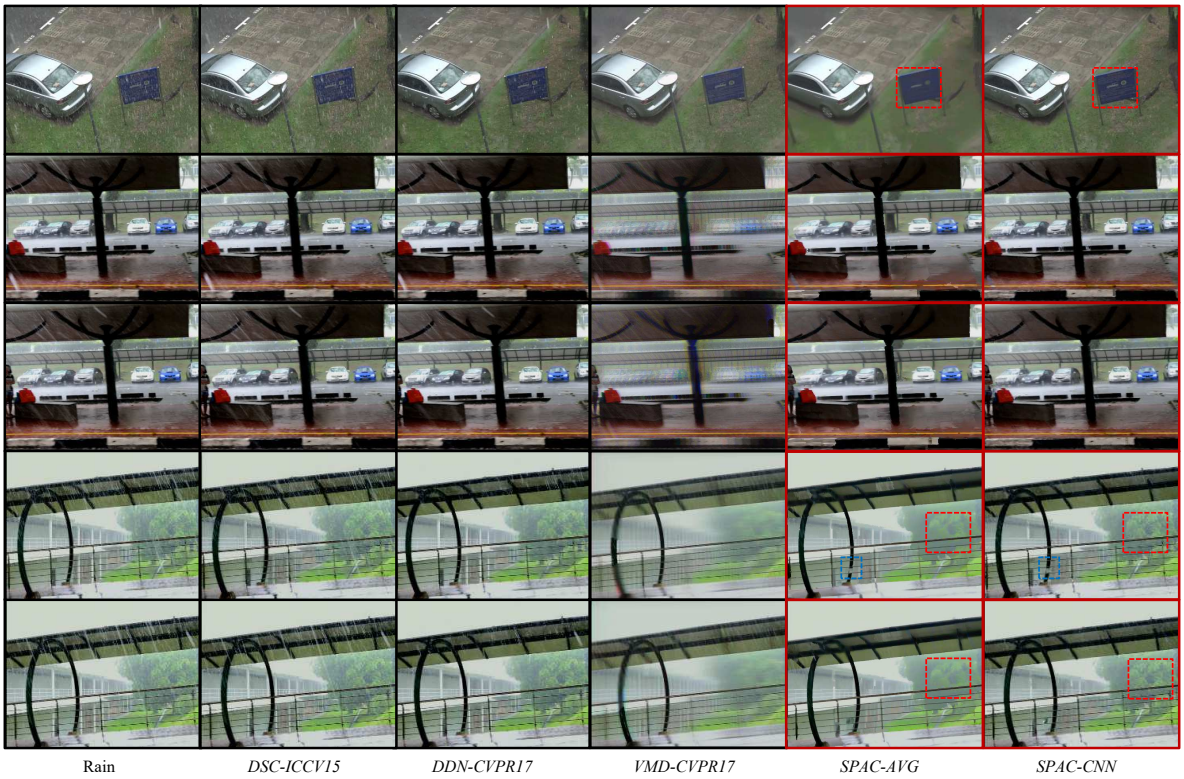


Figure 9: Visual comparison for different rain removal methods on real world rain data.

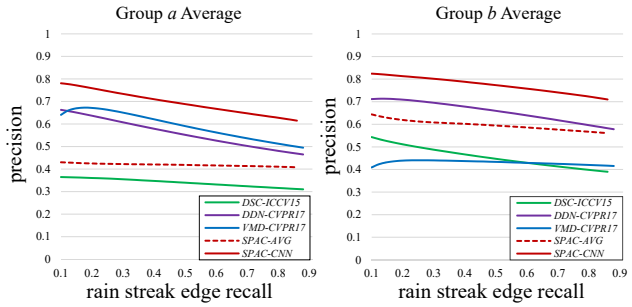


Figure 10: Rain edge pixel detection precision-recall curves for different rain removal methods.

the labeling of each testing scene are shown in Fig. 7.

Four state-of-the-art methods were chosen for comparison: two image-based derain methods, i.e., discriminative sparse coding (*DSC*) [22], and the deep detail network (*DDN*) [8]; one video-based method via matrix decomposition (*VMD*) [24]. The intermediate derain output X_{Avg} is also used as a baseline (abbr. as *SPAC-Avg*).

4.1.1 Rain Streak Edge Precision Recall Rates

Rain fall introduces edges and textures over the background. To evaluate how much of the modifications from the derain algorithm contributes positively to ***only removing the rain pixels***, we calculated the rain streak edge precision-recall (PR) curves. Absolute difference values were calculated between the derain output against the scene ground truth. Different threshold values were applied to retrieve a set of binary maps, which were next compared against the ground truth rain pixel map to calculate the precision recall rates.

Average PR curves for the two groups of testing scenes by different algorithms are shown in Fig. 10. As can be seen, for both Group *a* and *b*, *SPAC-CNN* shows consistent advantages over *SPAC-Avg*, which proves that the CNN model can efficiently compensate scene content details and suppress influences from rain streak edges.

Video-based derain methods (i.e., *VMD* and *SPAC-CNN*) perform better than image-based methods (i.e., *DSC* and *DDN*) for scenes in Group *a*. With slow camera motion, temporal correspondence can be accurately established, which brings great advantage to video-based methods. However, with fast camera motion, the performance of *VMD* deteriorates seriously for Group *b* data: rain removal is now at the cost of background distortion. Image-based methods show its relative efficiency in this scenario. However, *SPAC-CNN* still holds advantage over image-based methods at all recall rates for Group *b* data, which shows its robustness for fast moving camera.

4.1.2 Scene Reconstruction PSNR/SSIM

We calculated the reconstruction PSNR/SSIM between different state-of-the-art methods against the ground truth, and the results are shown in Table 1. The F-measure for rain

Table 2: Derain PSNR (*dB*) with different features absent.

Clip No.	$\hat{\mathcal{F}}_2$+$\hat{\mathcal{F}}_3$ (w/o $\hat{\mathcal{F}}_1$)	$\hat{\mathcal{F}}_1$+$\hat{\mathcal{F}}_3$ (w/o $\hat{\mathcal{F}}_2$)	$\hat{\mathcal{F}}_1$+$\hat{\mathcal{F}}_2$ (w/o $\hat{\mathcal{F}}_3$)	$\hat{\mathcal{F}}_1$+$\hat{\mathcal{F}}_2$+$\hat{\mathcal{F}}_3$
*a*1	25.28	28.87	27.63	**29.78**
*b*4	28.62	31.97	32.99	**33.79**

streak edge PR curves are also listed for each data.

As can be seen, *SPAC-CNN* is consistently 5 *dB* higher than *SPAC-Avg* for both Groups *a* and *b*. SSIM is also at least 0.06 higher. This further validates the efficiency of the CNN detail compensation network.

Video based methods (*VMD* and *SPAC-CNN*) show great advantages over image-based methods for Group *a* data (around 2dB and 5dB higher respectively than *DSC*). For Group *b*, image-based methods excel *VMD*, however *SPAC-CNN* still hold a 3dB advantage over *DDN*, 4dB over *DSC*.

4.1.3 Feature Evaluation

We evaluated the roles different input features play in the final derain PSNR over two testing data *a*1 and *b*4. Three baseline CNNs with different combinations of features as input were independently trained for this evaluation. As can be seen from the results in Table. 2, combination of the three features $\hat{\mathcal{F}}_1 + \hat{\mathcal{F}}_2 + \hat{\mathcal{F}}_3$ provides the highest PSNR. $\mathcal{F}_1$ proves to be the most important feature. Visual inspection on the derain output show both $\hat{\mathcal{F}}_2$+$\hat{\mathcal{F}}_3$ and $\hat{\mathcal{F}}_1$+$\hat{\mathcal{F}}_3$ leaves significant amount of un-removed rain. Comparing the last two columns, it can be seen that $\hat{\mathcal{F}}_3$ works more efficiently with *a*1 than *b*4, which makes sense since the high frequency features are better aligned for slow cameras, which led to more accurate detail compensation.

4.2. Visual Comparison

We carried out visual comparison to examine the derain performance of different algorithms. Fig. 8 shows the derain output for the testing data *a*.3, *b*.1, and *b*.4. Two consecutive frames are shown for *b*.1 and *b*.4 to demonstrate the camera motion. As can be seen, image-based derain methods can only handle well light and transparent rain occlusions. For those opaque rain streaks that cover a large area, they fail unavoidably. Temporal information proves to be critical in truthfully restoring the occluded details.

It is observed that rain can be much better removed by video-based methods. However the *VMD* method creates serious blur when the camera motion is fast. The derain effect for *SPAC-CNN* is the most impressive for all methods. The red dotted rectangles showcase the restored high frequency details between *SPAC-CNN* and *SPAC-Avg*.

Although the network has been trained over synthetic rain data, experiments show that it generalizes well to real world rain. Fig. 9 shows the derain results. As can be seen, the advantage of *SPAC-CNN* is very obvious under heavy rain, and robust to fast camera motion.

Table 3: Execution time (in *sec*) comparison for different methods on deraining a single VGA resolution frame.

DSC [22]	*DDN* [8]	*VMD* [24]	*SPAC-CNN*	
			SPAC-Avg	CNN
Matlab	Matlab	Matlab	C++	Matlab
236.3	0.9	119.0	0.2	3.1

4.3. Execution Efficiency

We compared the average runtime between different methods for deraining a VGA resolution frame. Results are shown in Table 3. As can be seen *SPAC-Avg* is much faster than all other methods. *SPAC-CNN* is much faster than video-based method, and it's comparable to that of *DDN*.

5. Discussion

For *SPAC-CNN*, the choice of SP as the basic operation unit is key to its performance. When other decomposition units are used instead (e.g., rectangular), matching accuracy deteriorates, and very obvious averaging blur will be introduced especially at object boundaries.

Although the SP template matching can only handle translational motion, alignment errors caused by other types of motion such as rotation, scaling, and non-ridge transforms can be mitigated with global frame alignment before they are buffered (as shown in Fig. 2) [27]. Furthermore, these errors can be efficiently compensated by the CNN.

When camera moves even faster, SP search range n_s needs to be enlarged accordingly, which increases computation loads. We have tested scenarios with camera speed going up to 50 *km/h*, the PSNR becomes lower due to larger mis-alignment blur, alignment error is also possible as showcased in blue rectangles in Fig. 9. We believe a re-trained CNN with training data from such fast moving camera will help improve the performance.

6. Conclusion

We have proposed a video-based rain removal algorithm that can handle torrential rain fall with opaque streak occlusions from a fast moving camera. SP have been utilized as the basic processing unit for content alignment and occlusion removal. A CNN has been designed and trained to efficiently compensate the mis-alignment blur introduced by deraining operations. The whole system shows its efficiency and robustness over a series of experiments which outperforms state-of-the-art methods significantly.

Acknowledgment

The research was partially supported by the ST Engineering-NTU Corporate Lab through the NRF corporate lab@university scheme.

References

[1] *Adobe After Effects Software*. Available at www.adobe.com/AfterEffects.

[2] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. Susstrunk. SLIC superpixels compared to state-of-the-art superpixel methods. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(11):2274–2282, 2012.

[3] P. C. Barnum, S. Narasimhan, and T. Kanade. Analysis of rain and snow in frequency space. *International Journal of Computer Vision*, 86(2):256, Jan 2009.

[4] H. Bay, A. Ess, T. Tuytelaars, and L. V. Gool. Speeded-Up Robust Features (SURF). *Computer Vision and Image Understanding*, 110(3):346 – 359, 2008.

[5] J. Bossu, N. Hautière, and J.-P. Tarel. Rain or snow detection in image sequences through use of a histogram of orientation of streaks. *International Journal of Computer Vision*, 93(3):348–367, Jul 2011.

[6] D.-Y. Chen, C.-C. Chen, and L.-W. Kang. Visual depth guided color image rain streaks removal using sparse coding. *IEEE Transactions on Circuits and Systems for Video Technology*, 24(8):1430–1455, Aug. 2014.

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

[8] X. Fu, J. Huang, D. Z. Y. Huang, X. Ding, and J. Paisley. Removing rain from single images via a deep detail network. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2017.

[9] K. Garg and S. K. Nayar. Detection and removal of rain from videos. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 1, pages 528–535, 2004.

[10] K. Garg and S. K. Nayar. When does a camera see rain? In *IEEE International Conference on Computer Vision*, volume 2, pages 1067–1074, Oct 2005.

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

[12] X. Glorot and Y. Bengio. Understanding the difficulty of training deep feedforward neural networks. In *International Conference on Artificial Intelligence and Statistics*, pages 249–256, 2010.

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

[14] J. Kim, J. K. Lee, and K. M. Lee. Accurate image super-resolution using very deep convolutional networks. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1646–1654, June 2016.

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

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

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

[18] T. G. Kolda and B. W. Bader. Tensor decompositions and applications. *SIAM review*, 51(3):455–500, 2009.

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

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

[21] Z. Li and J. Chen. Superpixel segmentation using linear spectral clustering. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1356–1363, June 2015.

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

[23] S. G. Narasimhan and S. K. Nayar. Vision and the atmosphere. *International Journal of Computer Vision*, 48(3):233–254, 2002.

[24] W. Ren, J. Tian, Z. Han, A. Chan, and Y. Tang. Video desnowing and deraining based on matrix decomposition. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 4210–4219, 2017.

[25] V. Santhaseelan and V. K. Asari. A phase space approach for detection and removal of rain in video. In *Intelligent Robots and Computer Vision XXIX: Algorithms and Techniques*, volume 8301, page 830114. International Society for Optics and Photonics, Jan. 2012.

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

[27] C.-H. Tan, J. Chen, and L.-P. Chau. Dynamic scene rain removal for moving cameras. In *IEEE International Conference on Digital Signal Processing*, pages 372–376. IEEE, 2014.

[28] P. H. Torr and A. Zisserman. MLESAC: a new robust estimator with application to estimating image geometry. *Computer Vision and Image Understanding*, 78(1):138–156, 2000.

[29] A. Tripathi and S. Mukhopadhyay. Video post processing: low-latency spatiotemporal approach for detection and removal of rain. *IET Image Processing*, 6(2), 2012.

[30] M. Van den Bergh, X. Boix, G. Roig, B. de Capitani, and L. Van Gool. *SEEDS: Superpixels Extracted via Energy-Driven Sampling*, pages 13–26. Springer Berlin Heidelberg, 2012.

[31] A. Vedaldi and K. Lenc. Matconvnet: Convolutional neural networks for matlab. In *ACM International Conference on Multimedia*, MM '15, pages 689–692, 2015.

[32] W. Yang, R. T. Tan, J. Feng, J. Liu, Z. Guo, and S. Yan. Deep joint rain detection and removal from a single image. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1357–1366, 2017.

[33] S. You, R. T. Tan, R. Kawakami, Y. Mukaigawa, and K. Ikeuchi. Adherent raindrop modeling, detectionand removal in video. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 38(9):1721–1733, Sept 2016.

[34] K. Zhang, W. Zuo, Y. Chen, D. Meng, and L. Zhang. Beyond a gaussian denoiser: Residual learning of deep cnn for image denoising. *IEEE Transactions on Image Processing*, 26(7):3142–3155, July 2017.

[35] 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 International Conference on Multimedia and Expo*, pages 461–464, July 2006.

[36] X. Zheng, Y. Liao, W. Guo, X. Fu, and X. Ding. Single-image-based rain and snow removal using multi-guided filter. In *International Conference on Neural Information Processing*, pages 258–265. Springer Berlin Heidelberg, 2013.