Controlling the Rain: from Removal to Rendering

Siqi Ni, Xueyun Cao, Tao Yue, Xuemei Hu
School of Electronic Science and Engineering, Nanjing University, Nanjing, China
nisiqi@smail.nju.edu.cn, caoxueyun@smail.nju.edu.cn, yuetao@nju.edu.cn, xuemeihu@nju.edu.cn

Abstract

Existing rain image editing methods focus on either removing rain from rain images or rendering rain on rain-free images. This paper proposes to realize continuous control of rain intensity bidirectionally, from clear rain-free to downpour image with a single rain image as input, without changing the scene-specific characteristics, e.g. the direction, appearance and distribution of rain. Specifically, we introduce a Rain Intensity Controlling Network (RICNet) that contains three sub-networks of background extraction network, high-frequency rain-streak elimination network and main controlling network, which allows to control rain image of different intensities continuously by interpolation in the deep feature space. The HOG loss and autocorrelation loss are proposed to enhance consistency in orientation and suppress repetitive rain streaks. Furthermore, a decremental learning strategy that trains the network from downpour to drizzle images sequentially is proposed to further improve the performance and speedup the convergence. Extensive experiments on both rain dataset and real rain images demonstrate the effectiveness of the proposed method.

1. Introduction

Transforming the weather condition of images realistically is a knotty but attractive problem. Specifically, transforming images between the rain and rain-free weather conditions, i.e., deraining [9,38,39] and rain rendering [10,12], enables us to remove rain from rain images or produce rain effect on rain-free images. However, all these methods focus on directly transforming from a certain weather condition to the other, without considering the bidirectional-controlling capability, leaving an innovative problem, i.e., continuously tuning the rain intensities bidirectionally (decreasing or increasing), ranging from rain-free scene to downpour with a single rain image as input.

In practice, the rains with different intensities exhibit much different characteristics in the density of rain streaks, the size of raindrops, and the thickness of fog. Meanwhile, other scene-specific characters are supposed to be preserved

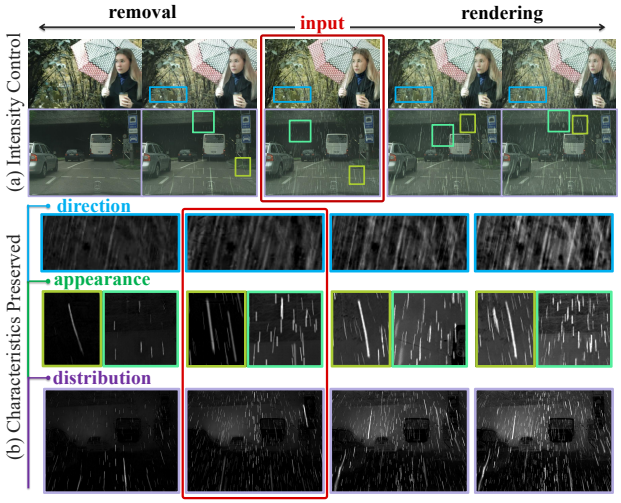

Figure 1. The effect of bi-directional rain controlling. (a) Continuous rain intensity control of the single rain image input (marked with red rectangle). (b) The preserved rain characteristics.

with the changing of rain intensity, including the direction of rain in local area, the appearance of raindrops, and the distribution, which mostly depend on the wind direction, illumination directions and camera parameters (e.g., focal length, point-of-focus and exposure time), as shown in Fig. 1. Different from the simple and naive method that removes the rain and renders new rain effect to form rain images with different rain intensities, our method can control the intensity-dependent characters changing with the rainfall, while preserving the scene-specific features of the input rain image, so that the generated rain images with different intensities look consistent with the input rain image.

In this paper, a Rain Intensity Controlling Network (RICNet) is proposed, which allows continuous and bidirectional control of the rain from removal to rendering with different intensities and similar scene-specific characteristics from a given rain image. RICNet is composed of three sub-networks, i.e, Background Extraction Network (BEN), High-frequency Rain-streak Elimination Network (HREN) and Main Controlling Network (MCN). Facilitated by the background and fog related information extracted by BEN and HREN, MCN could generate rain images of dif-

ferent intensities continuously and bidirectionally with the dual control of Parallel Gating Module (PGM) and Noise Boosting Module (NBM), based upon a multi-level interpolation framework. To preserve the scene-specific rain characteristics and reduce the repetition of rain streaks generated by the rendering network, histogram of oriented gradient and autocorrelation loss are proposed. Besides, a decremental training strategy that trains the network from downpour to drizzle images sequentially is also used to further improve the performance of proposed method.

In particular, the main contributions of this paper are:

- We *propose* a rain intensity controlling network to control the rain continuously in a bi-directional manner and preserve the scene-specific rain characteristics.
- We *introduce* the histogram of oriented gradient loss and autocorrelation loss to preserve the directions of rain streaks and suppress repetitive rain streaks. Decremental training strategy is further *designed* to improve the performance and speed up the convergence.
- We *conduct* extensive experiments on both synthetic rain dataset and real rain images, and demonstrate the effectiveness of the proposed method.

2. Related Work

Deraining Traditional deraining methods [9,21,22,34,37, 39] remove the rain with different intensities indiscriminately, which may lead to over/under-deraining for inputs with large intensity difference. Yasarla *et al*. [37] construct an uncertainty guided deraining network. Li *et al*. [21] propose a two-stage module for heavy rain removal. Recently, rain intensity is considered in some deraining work [35,38]. Zhang *et al*. [38] build a raindrop-density classifier to estimate the rainfall. Yang *et al*. [35] build a joint rain removal network to jointly estimate the rain densities and derain. These approaches could realize high quality rain removal, while fail to control the rainfall with different intermediate intensities. Multi-stage based methods [8, 15, 23, 26, 28] remove the rain progressively, while the outputs of middle stages are the intermediate results of deraining, which do not correspond to rain images with intermediate intensities. In this paper, we propose a continuous rain control network, which could generate rain images of different intensities. With the proposed method, we can not only remove the rain in the rain image, but also generate rain images of intermediate intensities in a continuous way.

Rain Effect Rendering Approaches in different areas have been proposed to generate rain images. Garg and Nayar [10] generate realistic rain streaks based on the analysis of rain appearance model. Halder *et al*. [12] design a physics-based approach to render different levels of rain. However, these rendering works highly depend on the physical parameters, e.g., scene structure, illumination condition, wind speed and direction, as well as the camera settings, inapplicable to render rain images while preserving the scene-specific features from the input rain images. In this paper, we propose a rain control neural network that could render the rain with different intensities while preserving the scene-specific features.

Feature Interpolation Feature interpolation methods allow continuous image transform [13, 27, 30, 31] via interpolation in the feature space. He *et al*. [13] propose adaptive feature modification layers to modulate between two image restoration levels. Shoshan *et al*. [27] achieve interpolation by adding tuning-blocks to main network. Wang *et al*. [31] apply linear interpolation in the parameter space of networks. In addition, interpolation among multiple effects are proposed [4, 7, 14, 16, 24]. Huang *et al*. [14] feed the decoder with a set of weighted style features to achieve mixed styles. Li *et al*. [24] create new textures via interpolating corresponding selection units. Jing *et al*. [16] propose gating functions to alter multiple stroke sizes at one time. These methods can achieve general style transform, however, they can not be directly incorporated for different rain intensities transform since specific design for rain control scenarios should be considered. In this paper, we propose the RICNet to interpolate rain images with different intensities continuously with scene-specific characteristics.

3. Rain Intensity Controlling Network

The proposed RICNet is introduced in details including the network architecture, loss function and training strategy.

3.1. Network Architecture

As shown in Fig. 2, the overall architecture of RICNet is composed of three main sub-networks including background extraction network (BEN), high-frequency rain-streak elimination network (HREN), and main controlling network (MCN). Specifically, MCN plays the main role in controlling the rain with bi-directional scene-specific rain modulation and it is constructed with core modules of multi-level feature extraction, parallel gating module (PGM) and noise boosting module (NBM). BEN and HREN are proposed to provide the background and fog information to facilitate MCN in controlling the rain with clear background and realistic fog rendering.

BEN BEN is designed to extract the background information, and provide the intermediate information to MCN for rain rendering. It is designed based on generative adversarial network (GAN) [11]. The generator is constructed in a coarse-to-fine way. The prepositive coarse structure is designed to remove the rain coarsely, and the fine structure (U-net) fed with the coarse derained and original input is applied to refine the deraining. To generate rain images with sufficient background details, the features in the skip layers

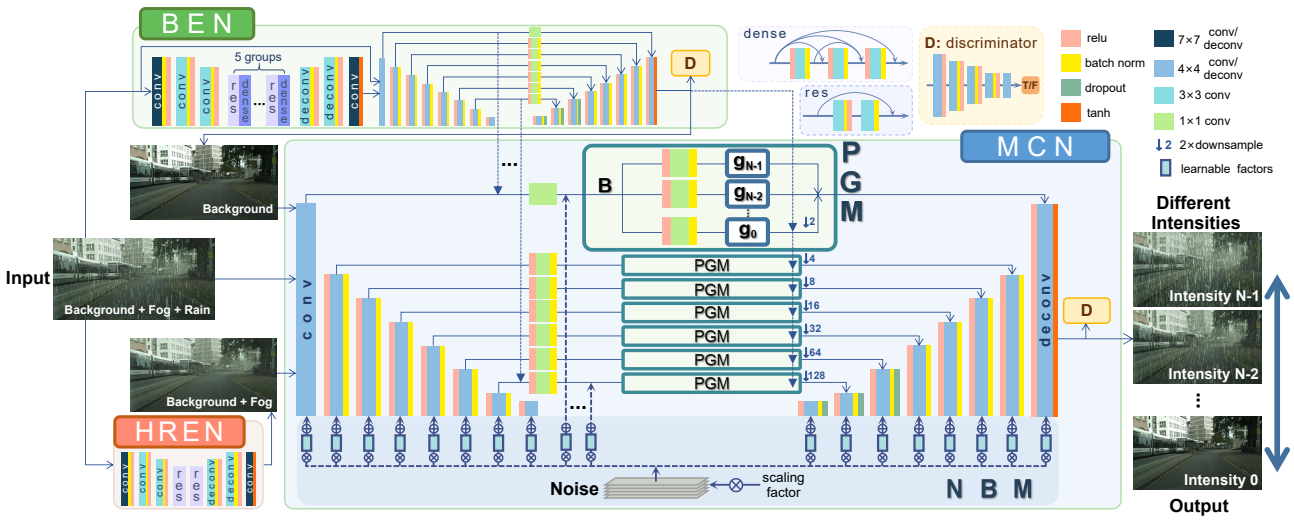

Figure 2. The overall architecture of RICNet, composed of BEN, HREN and MCN.

of the refining network of BEN are transmitted to the skip layers of MCN. Additionally, the output of BEN is not only sent to the input of MCN, but also the rain free branches (the branches used for generating the rain-free images) of the PGMs in MCN. The nearest-neighbor interpolation is used to downsample the output of BEN to the feature sizes of these branches. The specific way of information transmission is shown in Fig. 2. The discriminator is designed to evaluate the similarity between the output of generator and the desired rain-free image, as shown in Fig. 2.

HREN Since BEN is designed to extract the background information, both rain and fog information are removed. To control the rain rendering with scene-specific fog effect, we design HREN to eliminate the high frequency rain streaks in the input rain image and provide the image containing both background and fog information to MCN.

MCN MCN is the main network of rain control, which is also based on GAN [11]. The generator is composed of a multi-level interpolation U-net, PGM and NBM. The discriminator in MCN is designed to evaluate the similarity between the output of generator and the desired level of rain image, with the same discriminator architecture as BEN.

Considering that rain streaks distribute among a large range of scales, especially for different intensities of rain images. Existing feature interpolation methods [13, 27, 30] only consider interpolation at a single level, lacking capacities to utilize features at different scales. To build a network that could control the rain at different intensities, we introduce multi-level interpolation at the skip layers of different scale levels in the U-net network, as shown in Fig. 2.

PGM is proposed for interpolation, enabling the RICNet to be trained with discrete rain intensities and generate continuous intensities by weighting between the neighboring gates during testing. As shown in Fig. 2, PGM is introduced at the skip layers of different feature scales. Each PGM is composed of multiple branches targeting different rain intensities. The process of each PGM can be expressed as $f_{\mathrm{PGM}}(\boldsymbol{B}) = \sum_i g_i t_i(\boldsymbol{B}), \sum_i g_i = 1$, where $\boldsymbol{B}$ represents the rain features in the skip layer, $t(\cdot)$ denotes 1×1 convolution for turning $\boldsymbol{B}$ to the features for a specific rain intensity level. g_i is the gating parameter used to control the weighting degree of each branch. During training, N discrete intensity levels of output rain (i.e. $i = 0, 1, ..., N-1$) are controlled by binary gating parameters $\{g_i\}$ $(i \in 0, 1, ..., N-1)$. For each level of input rain, different levels of output rain are trained respectively, with only the i-th branch corresponding to the output level i opened (i.e. $g_i = 1$), and the rest closed. During testing, our RICNet can recognize the intensity of input rain automatically, and output user-specified intensity of rain by linearly combining the two neighboring branches using decimal gating parameters. Specifically, for any target rain intensity level s that is between two specific discrete levels, e.g. i-th and $i+1$-th, the output rain image can be obtained by

$$f^{\mathrm{test}}_{\mathrm{PGM}}(\boldsymbol{B}) = g_i t_i(\boldsymbol{B}) + g_{i+1} t_{i+1}(\boldsymbol{B}),$$
$$s.t. \quad g_i = \frac{l_{i+1} - l_s}{l_{i+1} - l_i}, g_{i+1} = \frac{l_s - l_i}{l_{i+1} - l_i}, \tag{1}$$

where l_i, l_{i+1} and l_s denote the corresponding rain intensity.

Inspired by the fact that noise can be effective in generating textures with randomness and high diversity [18, 24], we introduce NBM into MCN to further boost the distribution randomness and texture diversity of rain streaks. As shown in Fig. 2, zero-mean Gaussian noises are added to all pairs of encoder-decoder convolution layers and all skip layers in MCN. Furthermore, since scaling of noise intensity has already been demonstrated to be useful for the adjustment of interpolation effect [18, 24], we propose to control the scaling factor of noise to assist MCN in controlling the rain intensity. The noise image is firstly repeat-expanded

to the same channel as the features, and multiplied by the scaling parameter that corresponds to the target rain intensity level. Then each channel of the noise image is multiplied by a learnable factor, after which the weighted noise image is added to the features in MCN. During training, when the target output rain intensity changes from level 0 to $N-1$, the noise is multiplied with scaling parameter of $0, 1/(N-1), 2/(N-1), ..., 1$. In the testing process, to generate rain images of intensity level s between intensity level 0 and $N-1$, the noise is multiplied by $s/(N-1)$.

3.2. Loss Function

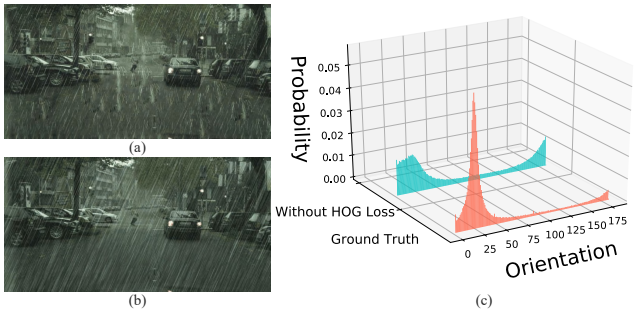

Figure 3. (a) Output rain image without HOG loss. (b) Ground truth rain image. (c) HOG comparison of (a) and (b).

HOG Loss To render different levels of rain, scene-specific physical properties such as the direction of rain should be retained. However, due to the difficulty in identifying various streak directions of different inputs, disorientation occurs, as shown in Fig. 3(a). In order to preserve the scene-specific direction information, we refer to the method of histogram of oriented gradient (HOG) [6], which is adopted to calculate the orientation of rain streaks in deraining studies [3, 32, 40]. Different from these methods, we propose to utilize HOG as one of the loss functions to enforce the same orientation distribution between the network output and the ground truth rain streaks, avoiding the inconsistency shown in Fig. 3(c).

In practice, we adopt Sobel operator to compute the horizontal and vertical gradient map (G_x, G_y). The gradient orientation is computed by $\theta = \tan^{-1}\frac{G_y}{G_x}$. Since the distribution of the direction of rain is mostly consistent in the local region, we divide the image into K cells of the same size. In each cell, we calculate the weighted histogram of orientation with orientation $[0, \pi)$ divided into M bins. The weight is the length of the gradient, i.e. $G = \sqrt{G_x^2 + G_y^2}$. For better invariance to noise and illumination, the original orientation histogram $\boldsymbol{O}_{\text{org}}$ is normalized with L1-norm, $\boldsymbol{O} = \boldsymbol{O}_{\text{org}} / \|\boldsymbol{O}_{\text{org}}\|_1$. We denote the normalized HOGs of ground truth and output in the k-th cell as $\boldsymbol{O}_k^{\text{gt}}$ and $\boldsymbol{O}_k^{\text{out}}$. The difference between the two distribution in the k-th cell is measured with Kullback Leibler (KL) divergence [20]. To measure the direction distribution similarities over the whole image, we adopt the mean of KL divergence of the K cells as the HOG loss in our network,

$$L_{\text{h}} = \frac{1}{K}\sum_k \text{KL}(\boldsymbol{O}_k^{\text{gt}}, \boldsymbol{O}_k^{\text{out}}). \tag{2}$$

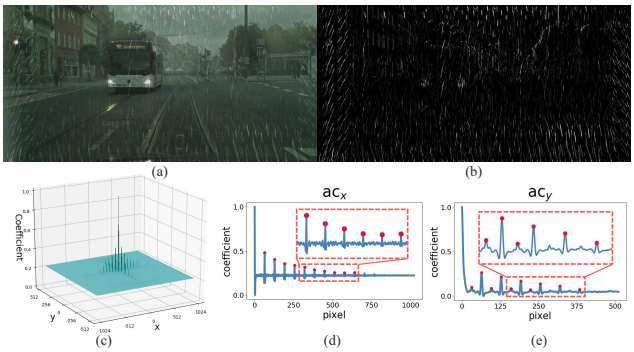

Figure 4. (a) Output rain image without autocorrelation loss. (b) The rain streak image of (a). (c) The 2-D autocorrelation map of (b). (d)(e) The autocorrelation along the x and y direction.

Autocorrelation Loss During experiments, we find that the generated rain streaks may repeat periodically in horizontal and vertical directions, as shown in Fig. 4. In order to solve this problem, we design a novel autocorrelation loss to eliminate the repetitive patterns. In order to reduce the influence of the background, especially the scenes with periodic patterns, rain mask is obtained by subtracting the background from the input rain image. Since rain streaks are near vertial strips, in this paper, we compute the autocorrelation of its horizontal gradient (shown in Fig. 4(b)).

In experiments, we observe that significant peaks only occur in the horizontal and vertical directions, so we only take the autocorrelation coefficients along the two directions (i.e., ac_x, ac_y) to construct the loss. To eliminate the scale difference between scenes, ac_x and ac_y are normalized to 0-1, as shown in Fig. 4(d)(e). As observed, peaks caused by the repetition of rain streaks are obvious (marked with red points). To reduce the repetition, the top P peaks are detected and penalized by our autocorrelation loss,

$$L_{\text{ac}} = \sum_{x,y}\sum_{i=1}^{P} \frac{\text{ac}_{x,y}(\max_i) - \text{mean}_{\text{ac}_{x,y}}}{P}, \tag{3}$$

where $\max_i$ represents the index of the top i-th coefficient. $\text{mean}_{\text{ac}_{\text{x,y}}}$ denote the mean values of ac_x and ac_y. Considering that the coefficients close to the zero point are high, which effects the calculation of mean value, the near-zero regions of ac_x and ac_y are not accounted for mean computation (the near-zero region is defined from zero point to the first local minimum).

Overall Loss The training process of RICNet contains two stages: I. First the BEN is trained for extracting the

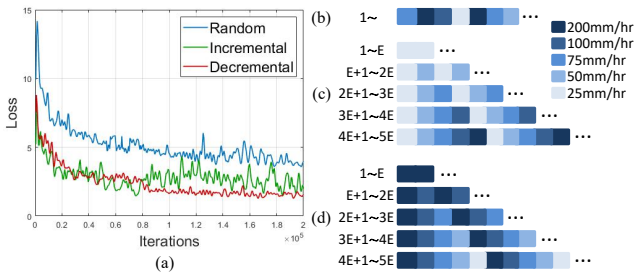

Figure 5. Illustration of training strategy. (a) Training curve comparisons among (b) random training, (c) incremental training and (d) our decremental training. E represents the epoch interval to update the rain intensity of the input samples.

background information. II. With the parameter of BEN fixed, HREN and MCN are trained together to generate clear foggy image and rain image of a certain level.

Specifically, in stage I, the coarse and fine networks in the generator of BEN are trained together, outputting rain-free images of $\boldsymbol{C}_1$ and $\boldsymbol{C}_2$, which are constrained with the GAN loss $L_{\mathrm{g}}^{\mathrm{I}}$, MAE loss L_{l1}, SSIM loss L_{s} [33], and perceptual loss L_{p} [17]. Denoting the ground truth background image as $\boldsymbol{C}_{\mathrm{gt}}$, the loss for training GAN in BEN is

$$\begin{aligned} L_G^{\mathrm{I}} = L_{\mathrm{g}}^{\mathrm{I}} &+ \lambda_{l1}^{\mathrm{I}} \left[L_{l1}(\boldsymbol{C}_1, \boldsymbol{C}_{\mathrm{gt}}) + L_{l1}(\boldsymbol{C}_2, \boldsymbol{C}_{\mathrm{gt}}) \right] \\ &- \lambda_{\mathrm{s}} \left[L_{\mathrm{s}}(\boldsymbol{C}_1, \boldsymbol{C}_{\mathrm{gt}}) + L_{\mathrm{s}}(\boldsymbol{C}_2, \boldsymbol{C}_{\mathrm{gt}}) \right] \\ &+ \lambda_{\mathrm{p}} \left[L_{\mathrm{p}}(\boldsymbol{C}_1, \boldsymbol{C}_{\mathrm{gt}}) + L_{\mathrm{p}}(\boldsymbol{C}_2, \boldsymbol{C}_{\mathrm{gt}}) \right], \end{aligned} \tag{4}$$

where $L_{\mathrm{g}}^{\mathrm{I}} = \log(1 - \mathrm{D}_{\mathrm{b}}(\boldsymbol{C}_2))$ represents the adversarial loss from discriminator D_{b}. The discriminator network D_{b} of BEN is trained with the discriminator loss [11], i.e. $L_D^{\mathrm{I}} = -\log(D_{\mathrm{b}}(\boldsymbol{C}_{\mathrm{gt}})) - \log(1 - D_{\mathrm{b}}(\boldsymbol{C}_2))$, where $\lambda_{l1}^{\mathrm{I}}$, λ_{s}, λ_{p} are the corresponding weighting coefficients.

In stage II, HREN and MCN are trained together and the corresponding generator loss is

$$\begin{aligned} L_G^{\mathrm{II}} &= L_{\mathrm{g}}^{\mathrm{II}} + \lambda_{\mathrm{h}} L_{\mathrm{h}}(\boldsymbol{I}, \boldsymbol{I}_{\mathrm{gt}}) + \lambda_{\mathrm{ac}} L_{\mathrm{ac}}(\boldsymbol{I}, \boldsymbol{I}_{\mathrm{gt}}) \\ &+ \lambda_{l1}^{\mathrm{II}} \left[L_{l1}(\boldsymbol{I}^{\mathrm{HREN}}, \boldsymbol{I}_{\mathrm{gt}}^{\mathrm{HREN}}) + L_{l1}(\boldsymbol{I}, \boldsymbol{I}_{\mathrm{gt}}) \right], \end{aligned} \tag{5}$$

where $L_{\mathrm{g}}^{\mathrm{II}} = \log(1 - D_{\mathrm{m}}(\boldsymbol{I}))$. $\boldsymbol{I}$ and $\boldsymbol{I}_{\mathrm{gt}}$ are the output of MCN and the ground truth rain image of a certain level. L_{h} and L_{ac} denote the HOG and autocorrelation loss. $\boldsymbol{I}^{\mathrm{HREN}}$ and $\boldsymbol{I}_{\mathrm{gt}}^{\mathrm{HREN}}$ denote the output and ground truth of HREN with only background and fog information. $\lambda_{l1}^{\mathrm{II}}$, λ_{h}, λ_{ac} are the weighting coefficients of the corresponding loss functions. The discriminator network D_{m} of MCN is trained with: $L_D^{\mathrm{II}} = -\log(D_{\mathrm{m}}(\boldsymbol{I}_{\mathrm{gt}})) - \log(1 - D_{\mathrm{m}}(\boldsymbol{I}))$.

3.3. Decremental Training Strategy

For training RICNet, we input the rain image samples of a certain intensity level i ($i \in \{0, ..., N-1\}$) and train the network to output rain images of different levels j ($j = \{0, ..., N-1\}$) of intensities (practical rainfall range from 0 mm/hr to 200 mm/hr). Empirically, we found that randomly selecting the rain intensities of the input samples for training yields inferior and oscillating performance, as shown in Fig. 5(a). To address the issue, we propose to train the network using ordered inputs to improve the effectiveness and robustness of the learning. In this paper, inspired by the curriculum learning [2], we propose a decremental training strategy that trains the samples with the largest rain intensity first, and adds a smaller rain intensity every E epochs. After all the intensities are involved in training, the network trains the samples randomly. We also compared the strategy with the random training and incremental training (which has been used in object detection with rain [12]), as show in Fig. 5(a), the proposed training strategy can greatly improve the convergence speed and final performance.

4. Experiment Results

4.1. Implement Details

Datasets In our experiments, we construct a dataset named RainLevel5 to train our RICNet as shown in Supplementary Fig. 2. The clean images are from Cityscapes dataset [5] and we synthesize 5 levels (25, 50, 75, 100 and 200mm/hr) of rain images with the methods in [1, 12]. The dataset contains 29750 rainy/clean image pairs, including 26870 pairs for training and 2880 pairs for testing. Besides, to simulate the rain realistically, we render the fog corresponding to different levels with the method in [12]. We further perform comparisons on a few synthetic datasets, i.e., Rain12000 [38], Rain800 [39], Rain200H [36], and real-world data from SPAData [29] (containing rainy/clean image pairs) and internet (without ground truth). Additionally, to control rain images with different hues, we randomly transform the rain images into the other hue during training of stage II, as is shown in Supplementary Fig. 3.

Training Parameters We adopt Adam [19] optimizer, with the learning rate initialized as 0.0002. BEN is trained with the learning rate divided by 2 after 10 epochs. The total training epoch of BEN is 15. HREN and MCN are trained for 6 epochs. In HOG loss, K and M are 512 and 180, and the patch size is 32×32. Larger K and M improve performance slightly at the cost of higher computation, while smaller values cause inferior results. In autocorrelation loss, P is 10, and smaller P fails at eliminating repetitiveness with large interval periods, while a very large P makes no sense since it is unlikely to appear a very large repetitive period in our experiment. Weighting coefficients $\lambda_{l1}^{\mathrm{I}}$, λ_{s}, λ_{p}, $\lambda_{l1}^{\mathrm{II}}$, λ_{h}, and λ_{ac} are set as 100, 100, 300, 100, 0.01 and 10 empirically. E is 1 and larger E causes overfitting of currently trained levels, which constraints the convergence of loss. The experiments are implemented on PyTorch [25] platform with an NVIDIA GeForce RTX 2080 GPU.

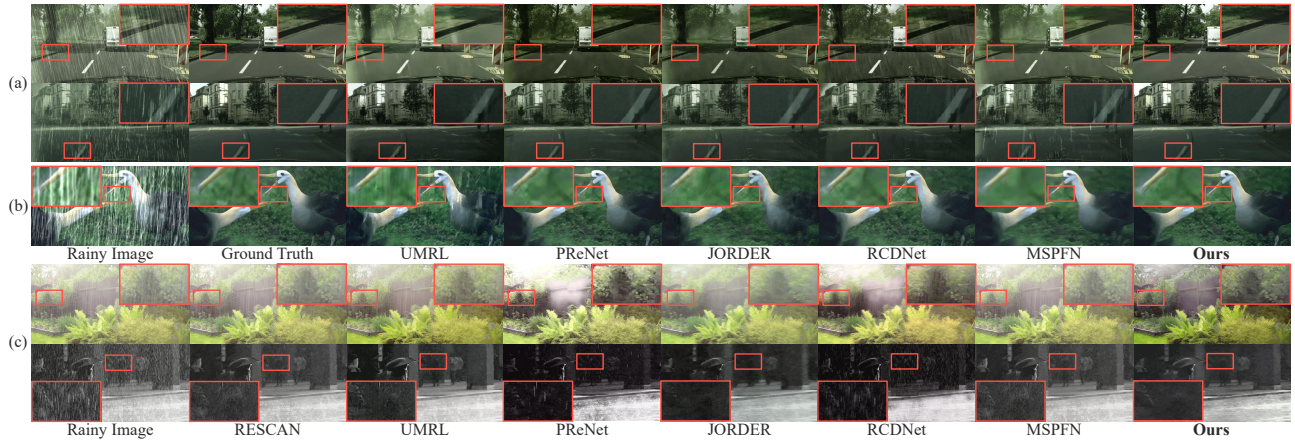

Figure 6. Deraining results on synthetic images of (a) RainLevel5, (b) Rain200H [36] and (c) real images.

Table 1. Derained comparisons on synthetic datasets (PSNR/SSIM).

Methods	RainLevel5	Rain12000	Rain800	Rain200H	SPAData
RESCAN [23]	24.09/0.777	30.51/0.882	25.00/0.835	28.02/0.862	38.11/0.971
UMRL [37]	23.13/0.804	29.77/**0.920**	24.41/0.829	27.06/0.847	35.06/0.941
PReNet [26]	32.07/0.979	30.03/0.889	24.81/0.851	28.56/0.880	40.16/0.982
SEMI [34]	21.08/0.727	26.05/0.822	22.35/0.788	22.17/0.719	35.31/0.941
JORDER [35]	32.88/0.965	24.32/0.862	26.73/0.869	23.45/0.749	40.78/0.981
HRGAN [21]	35.99/0.980	30.87/0.891	26.80/0.853	28.75/0.882	37.45/0.952
RCDNet [28]	21.34/0.860	26.44/0.816	23.75/0.842	28.82/0.893	**41.47/0.983**
MSPFN [15]	26.27/0.856	32.39/0.916	**27.50/0.876**	26.97/0.835	37.87/0.957
Ours	**37.81/0.985**	**32.67**/0.892	27.11/0.869	**28.84/0.893**	37.98/0.972

4.2. Rain Removal

We evaluate deraining performance of BEN in our RICNet on both synthetic and real data with the state-of-the-art (SOTA) deraining methods. The comparisons on synthetic dataset are shown in the Table 1 and Fig. 6(a)(b). For fair comparison, all models are re-trained on our RainLevel5. As shown, our method can handle different types of rain streaks, and recover the background details with higher quality. Though specific complexity of RainLevel5 (e.g. containing fog of different levels) limits the performance of some SOTA methods, they show good results in other datasets and our method performs comparably.

The comparisons on real data are shown in Fig. 6(c). As shown, our method could recover the background with high fidelity, while MSPFN [15] leaves some rain streaks unremoved and RCDNet [28] produces unnatural white patches in the background. Besides, our method has a good defogging effect, which could further improve the visual quality of the generated rain images. In addition, the ablation study in rain removal is conducted in Supplementary Table 1.

4.3. Rain Control

Experiments on Synthetic and Real Data The rain control results on synthetic data of RainLevel5 are shown in Fig. 7, which demonstrates the ability of our RICNet to both achieve bi-directional rain control and preserve scene-specific rain characteristics. With a rain image of any intensity as input (marked with yellow box in Fig. 7), our network can output rain images of both lower and higher intensities. By which case, bi-directional effects of controllable rain removal and rendering can be achieved. Additionally, scene-specific rain characteristics can also be preserved. The appearance of raindrops in transformed rain images are consistent with the input. The orientation and distribution of the input rain are also well learned and preserved. Since the intensity of fog and rain is highly related, as shown in Supplementary Fig. 2[1], the fog intensity is automatically adjusted during the control of the rain intensity.

We further demonstrate the bi-directional and continuous rain control capability of our method on the real rain images. As shown in Fig. 8, our RICNet can remove or generate the rain with arbitrary intensities and preserve the scene-specific rain characteristics as the input rain images. More control results are shown in Supplementary Fig. 6.

Moreover, continuous intermediate levels can also be achieved via setting the gating parameters g_i as discussed in Eq. (1). As shown in Fig. 9, with the intensities of input rain images set as 100 mm/hr (Fig. 9(a)) and 25 mm/hr (Fig. 9(b)), intermediate intensities of rain images can be generated between 100-200 mm/hr and 25-50 mm/hr. The interpolated results show continuous transformation and well-preserved characteristics with the input rain images.

Comparison with other Interpolation Methods To further demonstrate the interpolation effectiveness of our RICNet, we conduct comparisons with other interpolation methods [27,31]. Specifically, since [27] and [31] can only transform between two effects, we set the Effect 1 and Effect 2 as the generation of rain with intensity of 25mm/hr and 200mm/hr. As for [27], first the main network is trained to realize Effect 1. Then main network is fixed, and tuning-blocks are added and trained to realize Effect 2. During testing, the interpolation between Effect 1 and 2 can be

[1]Most of the fog in rain is composed of the distant raindrops, which cannot be seen clearly.

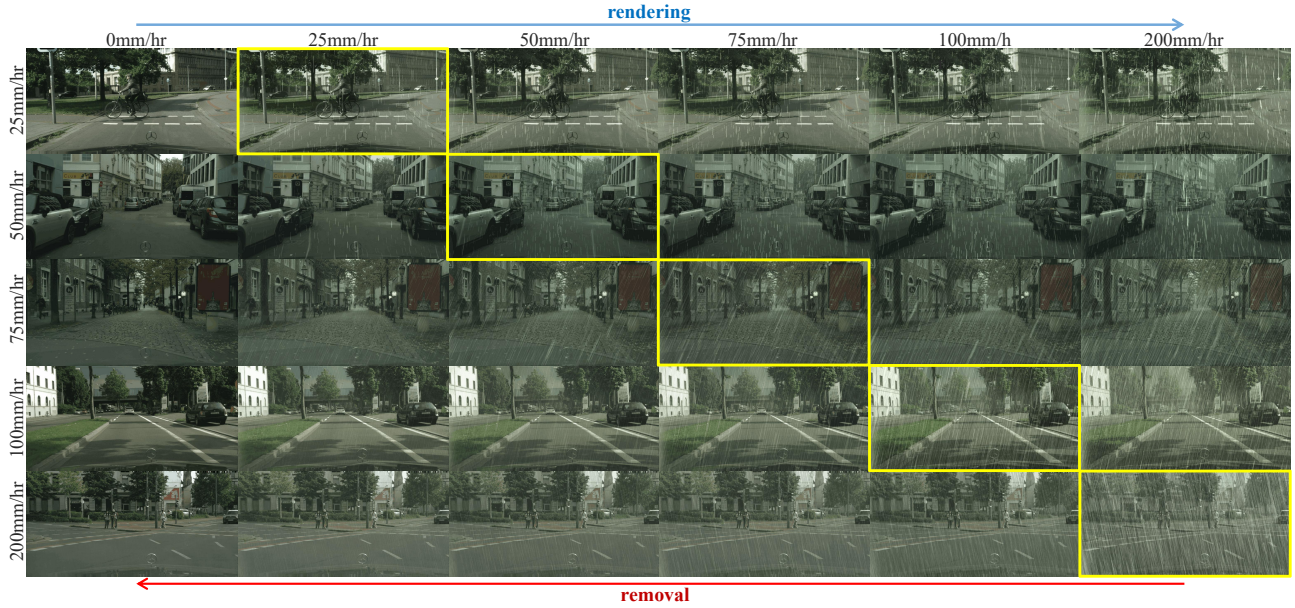

Figure 7. Rain control results on RainLevel5. Yellow boxes mark the input rain images and others are the generated rain images.

Figure 8. Bi-directional and continuous rain control results on real rain images.

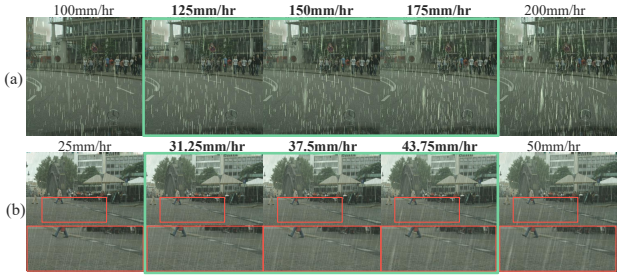

Figure 9. Continuous interpolation results within the range of rain intensities (a) 100 to 200mm/hr and (b) 25 to 50mm/hr. The interpolated rain images are marked with green boxes.

achieved by adjusting the participation degree of tuning-blocks. The interpolated results are shown in Fig. 10(b). As observed, since the interpolation is implemented in low-level feature space, the appearance of rain is not natural enough. As for [31], we train our network, without PGM and NBM, separately to realize Effect 1 and 2 respectively. Via interpolating the two network parameters, we can get the intermediate effects, as shown in Fig. 10(c). We can see that the rain intensity of the result is smaller than the target, and the rain appearance is also unnatural. In comparison, our method could enable more natural interpolation effects.

Ablation Study We conduct an ablation study to demonstrate the importance of multi-level interpolation. In comparison, we only interpolate features in low and high levels through only open the PGMs in skip layers of the first

Figure 10. Comparisons between (a) our and other interpolation methods, i.e. (b) [27] and (c) [31]. Input rain intensity: 50 mm/hr.

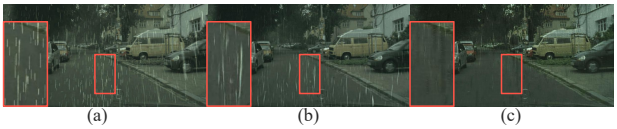

Figure 11. Results comparison of interpolation with (a) both low and high levels, (b) only low levels, and (c) only high levels.

4 pairs or the last 4 pairs of convolution layers of MCN. As shown in Fig. 11, large rain streaks tend to be distorted with only low-level interpolation (Fig. 11(b)). The network with only high-level interpolation fails to generate small rain streaks (Fig. 11(c)) and produces few artifacts. By contrast, the result with both low and high levels interpolation (Fig. 11(a)) is better at generating different scales of rain streaks, with a natural visual quality.

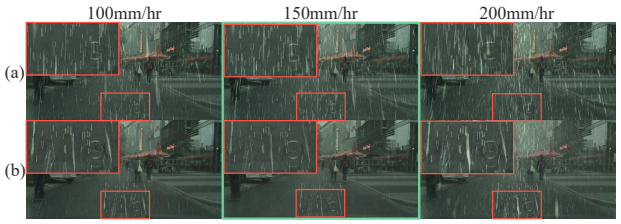

Figure 12. Comparison between (a) with and (b) without NBM.

We further study the effectiveness of the noise boosting module (NBM) in MCN. As shown in Fig. 12, the interpolated images (marked with the green boxes) show more diversified streak texture and natural distribution with NBM. Besides, results with NBM show better consistency in the changing trend of rain intensity, which further demonstrate the effectiveness of NBM in the control of rain intensity.

To demonstrate the effectiveness of our HOG loss and autocorrelation loss, we train RICNet with or without the corresponding loss. Figs. 13(a)(b) show that the HOG loss could ensure to preserve the orientation distribution of rain. Figs. 13(c)(b) demonstrate the effect of autocorrelation loss in suppressing repetitive rain streaks.

The effectiveness of random, incremental (training with input samples from 25 mm/hr to 200 mm/hr) and our decremental (training with input samples from 200 mm/hr to 25 mm/hr) training strategies are compared. As shown in Fig. 14, both outputs of random and incremental training show unnatural rain appearance. In comparison, the result of our decremental training demonstrates its superiority in generating more similar rain appearance with the input.

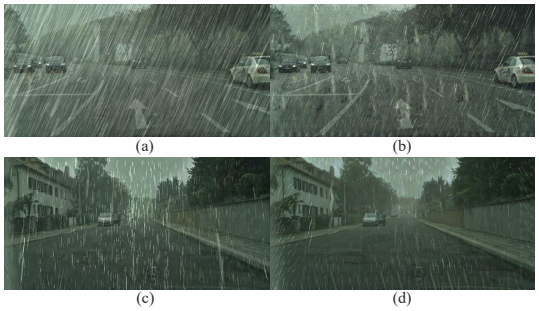

Figure 13. Comparison of results w/o HOG loss and autocorrelation loss. (a) With HOG Loss, (b) without HOG Loss, (c) with Autocorrelation Loss, and (d) without Autocorrelation Loss.

Figure 14. Comparison of training strategies. (a) Input rain image (50mm/hr), (b)-(d) results of output rain images at 87.5mm/hr with random, incremental and decremental training strategies.

We conduct numerical comparisons of rain rendering techniques in Table. 2. As shown, each technique helps enhance the perceptual effect and feature similarity, and some techniques (e.g., HOG loss, multi-level interpolation) show more contributions for boosting photorealism.

Table 2. Comparisons of rain rendering techniques.

Metric	**Ours**	Level		Without NBM	Loss		Training Strategies	
		only high	only low		without HOG	without autocorr	random	incremental
NIQE	3.985	5.132	4.974	4.955	5.612	4.842	4.975	5.031
FSIM	0.850	0.752	0.812	0.819	0.785	0.802	0.823	0.793

5. Discussion and Conclusion

In this paper, we propose RICNet to realize bi-directional and continuous rain control. Inputted with a rain image, RICNet could realize both removal and rendering of rain with scene-specific characteristics preserved. The HOG and autocorrelation loss are introduced to enhance the consistence in orientation and suppress repetitive rain streaks. To facilitate convergence with robustness and elegant performance, a decremental training strategy is introduced. Experiments on both synthetic and real data demonstrate the promising performance of the proposed method. Furthermore, our study provides a unique insight into generating rain based on the rain texture in input rain images, which provides a new perspective for rain rendering.

6. Acknowledgements

This work was supported by National Science Foundation of China (No. 61971465 and No. 61671236), and Fundamental Research Funds for the Central Universities of China (No. 0210-14380145 and 0210-14380155).

References

[1] http://www.photoshopessentials.com/photo-effects/rain/.

[2] Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. Curriculum learning. In *Proceedings of the Annual International Conference on Machine Learning*, pages 41–48, 2009.

[3] Jérémie Bossu, Nicolas Hautière, and Jean-Philippe 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, 2011.

[4] Dongdong Chen, Lu Yuan, Jing Liao, Nenghai Yu, and Gang Hua. Stylebank: An explicit representation for neural image style transfer. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1897–1906, 2017.

[5] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3213–3223, 2016.

[6] Navneet Dalal and Bill Triggs. Histograms of oriented gradients for human detection. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 1, pages 886–893. IEEE, 2005.

[7] Vincent Dumoulin, Jonathon Shlens, and Manjunath Kudlur. A learned representation for artistic style. *Proceedings of International Conference on Learning Representations*, 2017.

[8] Zhiwen Fan, Huafeng Wu, Xueyang Fu, Yue Huang, and Xinghao Ding. Residual-guide network for single image deraining. In *Proceedings of the ACM International Conference on Multimedia*, pages 1751–1759, 2018.

[9] Xueyang Fu, Jiabin Huang, Delu Zeng, Yue Huang, Xinghao Ding, and John Paisley. Removing rain from single images via a deep detail network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3855–3863, 2017.

[10] Kshitiz Garg and Shree K Nayar. Photorealistic rendering of rain streaks. *ACM Transactions on Graphics (TOG)*, 25(3):996–1002, 2006.

[11] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Proceedings of the Advances in Neural Information Processing Systems*, pages 2672–2680, 2014.

[12] Shirsendu Sukanta Halder, Jean-François Lalonde, and Raoul de Charette. Physics-based rendering for improving robustness to rain. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 10203–10212, 2019.

[13] Jingwen He, Chao Dong, and Yu Qiao. Modulating image restoration with continual levels via adaptive feature modification layers. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 11056–11064, 2019.

[14] Xun Huang and Serge Belongie. Arbitrary style transfer in real-time with adaptive instance normalization. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1501–1510, 2017.

[15] Kui Jiang, Zhongyuan Wang, Peng Yi, Chen Chen, Baojin Huang, Yimin Luo, Jiayi Ma, and Junjun Jiang. Multi-scale progressive fusion network for single image deraining. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8346–8355, 2020.

[16] Yongcheng Jing, Yang Liu, Yezhou Yang, Zunlei Feng, Yizhou Yu, Dacheng Tao, and Mingli Song. Stroke controllable fast style transfer with adaptive receptive fields. In *Proceedings of the European Conference on Computer Vision*, pages 238–254, 2018.

[17] Justin Johnson, Alexandre Alahi, and Li Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. In *Proceedings of the European Conference on Computer Vision*, pages 694–711. Springer, 2016.

[18] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4401–4410, 2019.

[19] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[20] Solomon Kullback and Richard A Leibler. On information and sufficiency. *The Annals of Mathematical Statistics*, 22(1):79–86, 1951.

[21] Ruoteng Li, Loong-Fah Cheong, and Robby T Tan. Heavy rain image restoration: Integrating physics model and conditional adversarial learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1633–1642, 2019.

[22] Ruoteng Li, Robby T Tan, and Loong-Fah Cheong. All in one bad weather removal using architectural search. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3175–3185, 2020.

[23] Xia Li, Jianlong Wu, Zhouchen Lin, Hong Liu, and Hongbin Zha. Recurrent squeeze-and-excitation context aggregation net for single image deraining. In *Proceedings of the European Conference on Computer Vision*, pages 254–269, 2018.

[24] Yijun Li, Chen Fang, Jimei Yang, Zhaowen Wang, Xin Lu, and Ming-Hsuan Yang. Diversified texture synthesis with feed-forward networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3920–3928, 2017.

[25] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. 2017.

[26] Dongwei Ren, Wangmeng Zuo, Qinghua Hu, Pengfei Zhu, and Deyu Meng. Progressive image deraining networks: A better and simpler baseline. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3937–3946, 2019.

[27] Alon Shoshan, Roey Mechrez, and Lihi Zelnik-Manor. Dynamic-net: Tuning the objective without re-training for synthesis tasks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3215–3223, 2019.

[28] Hong Wang, Qi Xie, Qian Zhao, and Deyu Meng. A model-driven deep neural network for single image rain removal. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3103–3112, 2020.

[29] Tianyu Wang, Xin Yang, Ke Xu, Shaozhe Chen, Qiang Zhang, and Rynson WH Lau. Spatial attentive single-image deraining with a high quality real rain dataset. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12270–12279, 2019.

[30] Wei Wang, Ruiming Guo, Yapeng Tian, and Wenming Yang. Cfsnet: Toward a controllable feature space for image restoration. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4140–4149, 2019.

[31] Xintao Wang, Ke Yu, Chao Dong, Xiaoou Tang, and Chen Change Loy. Deep network interpolation for continuous imagery effect transition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1692–1701, 2019.

[32] Yinglong Wang, Shuaicheng Liu, Chen Chen, and Bing Zeng. A hierarchical approach for rain or snow removing in a single color image. *IEEE Transactions on Image Processing*, 26(8):3936–3950, 2017.

[33] Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13(4):600–612, 2004.

[34] Wei Wei, Deyu Meng, Qian Zhao, Zongben Xu, and Ying Wu. Semi-supervised transfer learning for image rain removal. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3877–3886, 2019.

[35] Wenhan Yang, Robby T Tan, Jiashi Feng, Zongming Guo, Shuicheng Yan, and Jiaying Liu. Joint rain detection and removal from a single image with contextualized deep networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 42(6):1377–1393, 2019.

[36] Wenhan Yang, Robby T Tan, Jiashi Feng, Jiaying Liu, Zongming Guo, and Shuicheng Yan. Deep joint rain detection and removal from a single image. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1357–1366, 2017.

[37] Rajeev Yasarla and Vishal M Patel. Uncertainty guided multi-scale residual learning-using a cycle spinning cnn for single image de-raining. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8405–8414, 2019.

[38] He Zhang and Vishal M Patel. Density-aware single image de-raining using a multi-stream dense network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 695–704, 2018.

[39] He Zhang, Vishwanath Sindagi, and Vishal M Patel. Image de-raining using a conditional generative adversarial network. *IEEE Transactions on Circuits and Systems for Video Technology*, 2019.

[40] Lei Zhu, Chi-Wing Fu, Dani Lischinski, and Pheng-Ann Heng. Joint bi-layer optimization for single-image rain streak removal. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2526–2534, 2017.