Received February 15, 2020, accepted February 23, 2020, date of publication February 27, 2020, date of current version March 10, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.2976761

GAN-Based Rain Noise Removal From Single-Image Considering Rain Composite Models

TAKURO MATSUI, (Student Member, IEEE), AND MASAAKI IKEHARA, (Senior Member, IEEE)
Department of Electronics and Electrical Engineering, Keio University, Yokohama 223-8522, Japan
Corresponding author: Takuro Matsui (matsui@tkhm.elec.keio.ac.jp)

ABSTRACT Under severe weather conditions, outdoor images or videos captured by cameras can be affected by heavy rain and fog. For example, on a rainy day, autonomous vehicles have difficulty determining how to navigate due to the degraded visual quality of images. In this paper, we address a single-image rain removal problem (de-raining). As compared to video-based methods, single-image based methods are challenging because of the lack of temporal information. Although many existing methods have tackled these challenges, they suffer from overfitting, over-smoothing, and unnatural hue change. To solve these problems, we propose a GAN-based de-raining method. The optimal generator is determined by experimental comparisons. To train the generator, we learn the mapping between rainy and residual images from the training dataset. Besides, we synthesize a variety of rainy images to train our network. In particular, we focus on not only the orientations and scales of rain streaks but also the rainy image composite models. Our experimental results show that our method is suitable for a wide range of rainy images. Our method also achieves better performance on both synthetic and real-world images than state-of-the-art methods in terms of quantitative and visual performances.

INDEX TERMS Generative adversarial network, single-image de-raining, deep learning, image restoration, residual learning.

I. INTRODUCTION

Image restoration and enhancement are of considerable practical concern. The number of outdoor vision systems, such as surveillance cameras and dashboard cameras, has been increasing in the past years. One major issue for autonomous navigation systems is to drive under bad-weather conditions. The presence of rain streaks and fog degrades the quality and clearness of images as shown in Fig. 1a. Effective rain removal (de-raining) methods are required in multiple types of practical situations. Different from common image de-noising tasks, the de-raining task is more challenging because rain have no fixed shapes and orientations. This paper is an extension of work originally presented in [1].

For the last few decades, several investigations have been conducted on removing rain noises from a rainy observation. De-raining methods can be categorized into two types: one is video-based methods and the other is single-image based methods. Video-based methods [2]–[4] utilize multiple frames to decompose an image into a rain layer and a background layer. As compared to the video-based methods, the single-image based methods are more ill-pose and difficult problem for the sake of lack of temporal data.

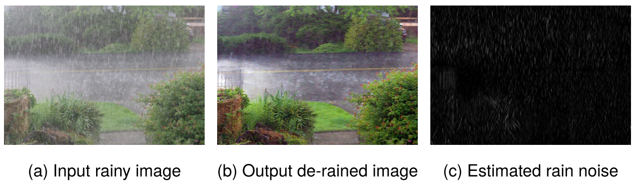
(a) Input rainy image (b) Output de-rained image (c) Estimated rain noise

FIGURE 1. A sample result of our proposed method for single image de-raining.

In this paper, we only address the single-image de-raining such as [5]–[15]. To our knowledge, Kang *et al.* [5] first tackle single-image de-raining based on morphological component analysis (MCA). To give other examples of conventional

The associate editor coordinating the review of this manuscript and approving it for publication was Shuping He.

methods, Luo *et al.* [6] introduce a discriminative sparse coding approach. Zhu *et al.* [7] propose three new priors defined by exploring local gradients. However, when the input image has structures similar with the rain streaks, this method tends to leave rain noises (under de-rain). Although Huang *et al.* [8] propose an image decomposition method based-on self-learning, it results in over-smoothing some regions (over de-rain). Recently, deep learning-based approaches have been proposed. In [11] by Fu *et al.*, it is observed that deep learning based methods are effective for de-raining. The authors train a three-layer convolutional neural network (CNN) on the detail (high-frequency) layer and combine the de-rained output with the enhanced base (low-frequency) layer. Although this approach is valid for de-raining, color of the output prones to be shifted due to the low-frequency layer. The rain structure in low-frequency component causes blur and haziness. Furthermore, because pixel values vary significantly throughout an image, learning the network is affected by the image background. Fu *et al.* extended the network structure using Res-block [16] in [12]. Yang *et al.* proposed a CNN structure that can jointly detect and remove rain streaks. Although their single CNN-based methods success in de-raining, they suffer from over de-rain and under de-rain. That is because simple CNN-based networks trained with MSE loss tend to generate blurry and unnatural results. To solve these problems, Zhang and Patel [13] propose a density aware GAN (Generative Adversarial Network) based method. Their method can clearly remove even heavy rain streaks. However, they tend to over smooth and lose important details. Thus, the trade-off between removing rain streaks and preserving textures is one of the most challenging problems.

In order to address these problems, we propose a GAN-based residual deep network. One example of our results is shown in Fig. 1. Residual learning can save the computational cost and success in several image tasks, such as an image recognition task [16] and an image de-noising task [17]. Our results show that residual learning achieves better performance than the image decomposition approach.

In addition, we focus on rainy image models. Our proposed method assumes two rainy image models as training dataset to make the network suitable to many real-world data. In [5], [18], the authors use a linear additive composite model (Fig. 2c) for synthesizing. Some recent studies, such as those conducted in [6], [11], [13], do not adopt the additive model but use a non-linear screen blend model (Fig. 2d). Because each model has both advantages and disadvantages, either is not always applicable to real-world images. Accordingly, we prepare training dataset which consists of images made by both the additive model and the screen blend model. Our experimental results demonstrate that this two-composite-model dataset is applicable to various real rainy images.

In summary, this paper makes the following contributions:

1) We explore an optimal deep learning structure for de-raining. Inspired by the success of GANs in other image processing tasks, we introduce a GAN for de-raining. Moreover, we compare the performance with several generator structures. Since a rain detection task need to capture global features and local features, we conclude that the UNet [19] structure is suitable for de-raining.
2) We introduce residual learning to remove rain streaks without losing the textures and edges. We learn the relationship between rainy images and residual images. Compared to a plane network structure which trains the mapping relationship between rainy images and clean images, clearer images are outputted. Also, This speeds up the training process and improves the de-raining performance. For real-world images, we do post-processing to remove haze.
3) To create synthetic rainy images, we introduce an automatic rain streaks generator. Almost all rain removal methods use Photoshop to create rain noises. As rain streaks have many parameters, automatically adjusting these parameters is quite difficult. Our proposed method can easily change parameters on Python, which results in saving time and effort to obtain natural rain streaks.
4) We propose combination of two composite models for creating synthetic rainy images. Although most existing methods use only one rain composite model, it is not enough for real-world images. Our experimental results demonstrate that a combination of these models achieves better performance than using either of them.

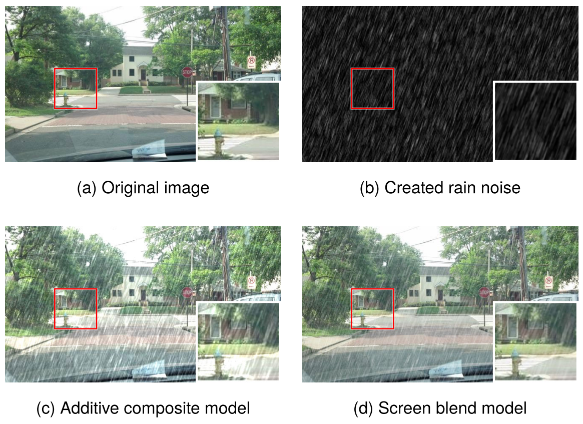

FIGURE 2. Comparison of two composite models.

This paper is organized as follows. Section II describes a brief review of de-raining and a few supporting methods related to our work. The details of our method are given in Section III. In Section IV, we present experimental results on both synthetic test data and real-world test data. Finally, Section V concludes the paper with a concise summary.

II. BACKGROUND AND RELATED WORKS

A. SINGLE-IMAGE RAIN REMOVAL

Unlike video-based de-raining [2]–[4], single-image de-raining is an extremely challenging task. That is because of its ill-posed nature and the unavailability of temporal information. In single-image rain removal, prior-based methods

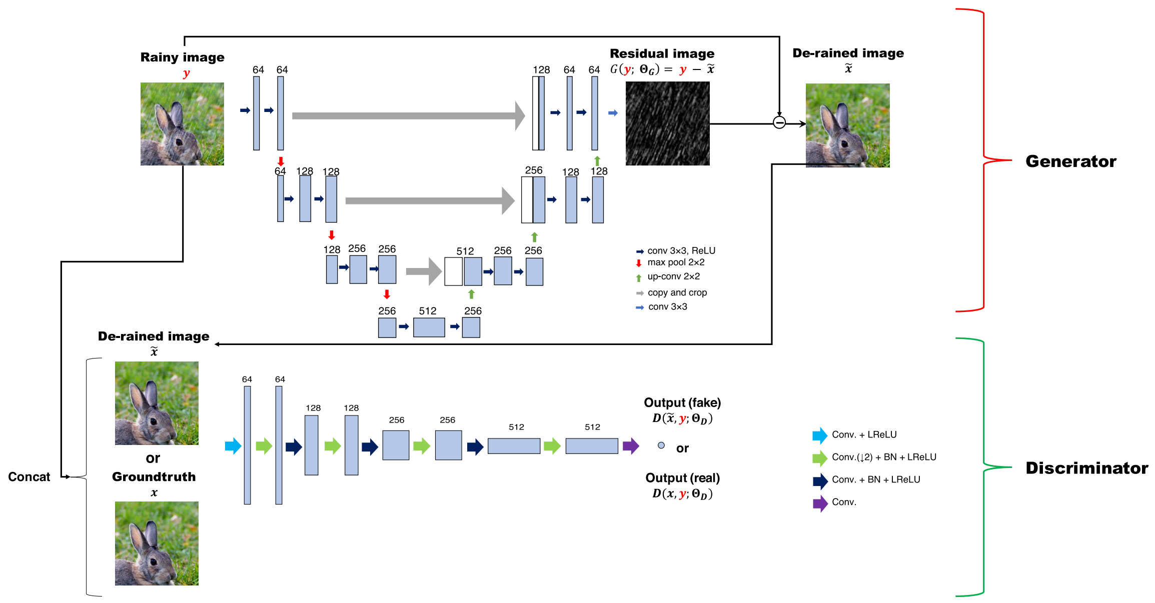

FIGURE 3. The proposed framework.

have been proposed. These include sparsity-based methods [5]–[7], low-rank representation-based methods [9] and Gaussian Mixture Model-based methods [10]. These prior-based methods tend to leave many rain streaks and over-smooth the details.

Recent studies introduce deep learning for single-image de-raining. These include simple CNN-based approaches [11], [12] and GAN based approaches [13]–[15], [20]. To train the network, synthetic rainy image dataset and corresponding clean image dataset are used.

B. GENERATIVE ADVERSARIAL NETWORK

In 2014, Goodfellow *et al.* [21] proposed a novel deep learning network called Generative Adversarial Network (GAN). Inspired by game theory, two neural networks contest with each other. One network called "generator" plays a role to generate new images from some input using the learned parameters. The other network called "discriminator" distinguishes whether the an input image is true of fake. GANs tend to be unstable during training and often produce unnatural artifacts. To solve these problems, many researchers have explored how to optimize GANs. In CGAN [22], conditional variables are introduced. Salimans *et al.* [23] propose mini batch discrimination to improve training. Creswell and Bharath [24] introduce a task specific cost function. In EBGAN [25], the discriminator is viewed as an energy function.

Now that GANs are attracting many researchers' interest, they use GANs to quite a few image processing tasks. For example, text-to-image synthesis [26], single image super-resolution [27], and image inpainting [28].

C. RAINY IMAGE COMPOSITE MODELS

In order to make synthetic rain images, several models have been proposed. The linear additive composite model in (1) is the simplest one. This is based on the hypothesis that rainy image $\boldsymbol{y}_{\text{add}}$ is broken down into background part $\boldsymbol{x}$ and rain streaks part $\boldsymbol{v}$:

$$\boldsymbol{y}_{\text{add}} = \boldsymbol{x} + \boldsymbol{v}. \tag{1}$$

As calculated values of $\boldsymbol{y}_{\text{add}}$ is clipped between 0 and 1 in the image processing, we can rewrite (1) as:

$$\boldsymbol{y}_{\text{add}} = \min(\boldsymbol{x} + \boldsymbol{v}, \mathbf{1}), \tag{2}$$

where $\mathbf{1}$ represents an array filled with ones and $\min(\boldsymbol{X}, \boldsymbol{Y})$ denotes the operation in which the smallest elements from $\boldsymbol{X}$ or $\boldsymbol{Y}$ are taken. This model tends to generate unnatural results when the original image contains bright region such as clouds or white objects (Fig. 2c). Otherwise, it is an optimal composite method for real-world extremely heavy rain.

On the other hand, a screen blend model is also adopted to synthesize images. In that model, to avoid unnatural appearance of the additive model, multiplied $\boldsymbol{x}$ and $\boldsymbol{v}$ are subtracted from the sum of them as:

$$\boldsymbol{y}_{\text{blend}} = \boldsymbol{x} + \boldsymbol{v} - \boldsymbol{x} \circ \boldsymbol{v}, \tag{3}$$

where $\circ$ indicates an element-wise multiplication operator. The images synthesized by the screen blend model look natural when an image contains shining part or dark part (Fig. 2d).

III. PROPOSED METHOD

We propose a GAN-based model in which the generator detects rain streaks and the discriminator judges whether the input clean image is a de-rained output (fake) or a clean image (true). The overall framework is illustrated in Fig. 3. While common CNN-based methods directly output de-rained images, the output of our generator is the rain streaks. This residual learning enhances the de-raining performance. In the training process, several number of rainy images are required. Since we do not have ground truth of real-world rainy images, we create pairs of synthetic rainy and clean image datasets. To be robust against many rain conditions, our synthetic rainy images are generated by two composite models.

A. THE NETWORK STRUCTURE OF PROPOSED GAN

During a test phase, in the input of our proposed network, the input is a rainy image $\boldsymbol{y}$ and the final output is a de-rained image $\tilde{\boldsymbol{x}}$ in Fig.3. Whereas, during the training phase, there are mainly four differences with the conventional methods

First, we do not use an image decomposition method. Many conventional methods divide the image into high frequency domain and low frequency domain, and each domain is processed independently. The problem is that they leave the rain noise in low frequency domain, which makes the restored image whitish.

Second, we use residual learning. In most CNN-based conventional methods, they directly output the de-rained image. The problem is that a tremendous amount of training dataset and training time are required to optimize the network. This is because it is difficult to recognize edges of the image and the rain streaks. Besides, the restoration of the contaminated image needs high computational cost. To solve these problems, we do not directly output a de-rained image $\tilde{\boldsymbol{x}}$ but output a residual image $\boldsymbol{y} - \tilde{\boldsymbol{x}}$ [1].

Third, we introduce UNet structure [19] to detect rain streaks. In order to distinguish rain streaks and background, both global features and local features have to be considered. Inspired by the success of UNet for image segmentation tasks [19], we adopt UNet architecture. In Table 1, we compare the quantitative results with the other deep learning networks: ResNet [16] and UNet++ [29]. The results demonstrate that UNet structure is optimal for de-raining.

Fourth, we propose a GAN structure. The discriminator identifies whether the input is a real clean image or a generated clean image. As can be seen in Fig. 4, the GAN-based network can suppress the artifacts (the upper image) and under de-rain (the lower image). Furthermore, to enhance the performance of the discriminator, we combine rainy image $\boldsymbol{y}$ with the de-rained image $\tilde{\boldsymbol{x}}$ as a fake data. Similarly, the real image is the rainy image $\boldsymbol{y}$ and the ground truth clean image $\boldsymbol{x}$.

B. LOSS FUNCTION

Our goal is to optimize parameters of the generator that minimize the following loss function:

$$\mathcal{L}_G = \mu_1 \mathcal{L}_{\text{MSE}} + \mu_2 \mathcal{L}_{\text{adv}}, \tag{4}$$

where μ_1 and μ_2 are the coefficients which are empirically determined. The first term, $\mathcal{L}_{\text{MSE}}$ is a mean square error between residual images $\boldsymbol{y_n} - \boldsymbol{x_n}$ and output images

TABLE 1. **Quantitative results for five test datasets using three different network structures. The number of epoch is 40 in each network. The upper part is PSNR (dB) and the lower part is SSIM. Red-letter and blue-letter values indicate the highest and the second highest values in each row, respectively.**

	Rainy image	ResNet	UNet	UNet++
Rain4	24.60	32.09	32.33	31.22
Rain12	28.82	32.48	31.38	32.67
Rain100	21.08	22.09	22.11	22.16
BSD100	23.31	29.55	30.76	29.01
Urban100	22.54	29.85	30.38	29.42
Rain4	0.834	0.958	0.958	0.945
Rain12	0.890	0.963	0.942	0.958
Rain100	0.760	0.819	0.812	0.821
BSD100	0.830	0.942	0.953	0.934
Urban100	0.822	0.950	0.955	0.946

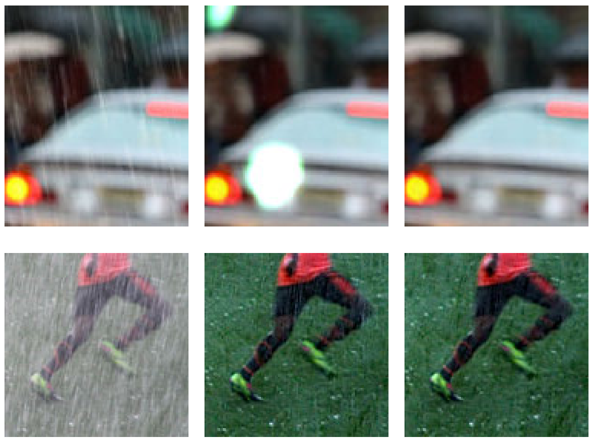
(a) Rainy image (b) UNet without discriminator (c) UNet with discriminator (proposed)

FIGURE 4. **Comparison of two different networks. Top: Synthetic image. Bottom: Real-world image. (a) rainy image, (b) de-rained output of UNet without discriminator, (c) de-rained output of UNet with discriminator (proposed GAN-based network).**

$G(\boldsymbol{y}_n; \boldsymbol{\Theta_G})$. The loss function is expressed as:

$$\mathcal{L}_{\text{MSE}} = \frac{1}{2N} \sum_{n=1}^{N} \|(\boldsymbol{y}_n - \boldsymbol{x}_n) - G(\boldsymbol{y}_n; \boldsymbol{\Theta_G})\|_2^2, \tag{5}$$

where n and N indicate an image index and the total number of images, respectively. $\boldsymbol{\Theta_G}$ denotes the learned parameters of the generator. The second term of the Eq. 4, $\mathcal{L}_{\text{adv}}$ is an adversarial loss. The loss function is to optimize the discriminator to evaluate whether the input is a true clean image or a fake clean image. The following equation is the adversarial loss.

$$\mathcal{L}_{\text{adv}} = \frac{1}{2N} \sum_{n=1}^{N} \|1 - D(\boldsymbol{x}_n, \boldsymbol{y}_n; \boldsymbol{\Theta_D})\|_2^2. \tag{6}$$

On the other hand, the discriminator parameters are optimized by minimizing the following loss function:

$$\mathcal{L}_D = \frac{1}{2N} \sum_{n=1}^{N} \left(\|D(\tilde{\boldsymbol{x}}, \boldsymbol{y}; \boldsymbol{\Theta_D})\|_2^2 + \|V - D(\boldsymbol{x}, \boldsymbol{y}; \boldsymbol{\Theta_D})\|_2^2 \right), \tag{7}$$

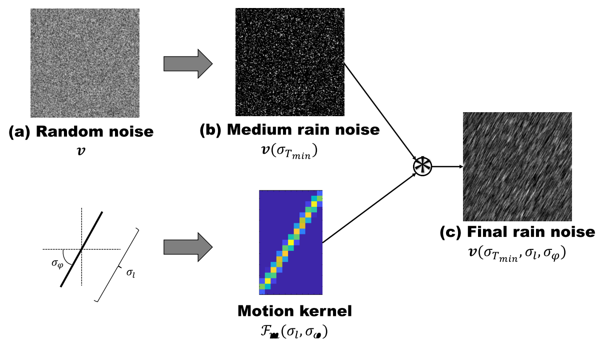

FIGURE 5. The flowchart of generating rain noises.

where V indicates a random value matrix which follows Gaussian distribution with an average of 1. To enable the discriminator trained effectively, we use random values instead of constant values.

C. GENERATING RAIN STREAKS

To train the mapping between rainy and clean images, we are required to create many natural rainy images from clean images. However, we cannot simultaneously get the same location images on a rainy day and a sunny day. Instead, we composite a clean image $\boldsymbol{x}$ and rain streaks $\boldsymbol{v}$ for training. Unlike the previous method as [11], [13] in which the authors synthesize rain streaks using Photoshop, we can easily generate rain noise $\boldsymbol{v}$ and control the densities and angles of rain on Python. As shown in Fig. 5, generating rain noise process follows three steps.

First, uniformly distributed random numbers $\boldsymbol{u} \in \mathcal{U}(0,1)$ are generated with the same size as the clean image. To shift the average value and normalize the number, we adjust the noise amount σ_a and clip them between 0 and 1. One element of the random noise $v_i \in \boldsymbol{v}$ is calculated as:

$$v_i \leftarrow \max\left(\min\left(\sigma_a(u_i - \lambda) + \lambda, 1\right), 0\right), \tag{8}$$

where $\lambda = 0.5$. Generated random noise is show in Fig. 5 (a). Next, the generated noise needs to be blurred by a Gaussian filter $\mathcal{F}_g$. Filtered output is normalized using values of $\sigma_{T_{\min}}$ and $\sigma_{T_{\max}}$. These two thresholds are used to reduce the amount of noise and boost its contrast. The calculated values are clipped between 0 and 1 as follows:

$$v_i \leftarrow \max\left(\min\left(\frac{\hat{v}_i - \sigma_{T_{\min}}}{\sigma_{T_{\max}} - \sigma_{T_{\min}}}, 1\right), 0\right), \tag{9}$$

where $\hat{\boldsymbol{v}} = \mathcal{F}_g \boldsymbol{v}$. The medium rain noise looks like Fig. 5 (b). Lastly, natural rain streaks are reproduced by giving the movement with a certain direction. After applying a motion filter $\mathcal{F}_m(\sigma_l, \sigma_\varphi)$, we adjust the rain scale σ_s. The filter $\boldsymbol{\mathcal{F}_m}$ is a function of the length σ_l and angle σ_φ.

$$\boldsymbol{v} \leftarrow \sigma_s \mathcal{F}_m(\sigma_l, \sigma_\varphi)\boldsymbol{v}. \tag{10}$$

The final rain noise is show in Fig. 5 (c). To simplify the noise model, we fix some parameters except for $\boldsymbol{v}$ as a function of $\sigma_{T_{\min}}$, σ_l and σ_φ. In the above procedure, we can obtain natural rain noise $\boldsymbol{v}(\sigma_{T_{\min}}, \sigma_l, \sigma_\varphi)$. Fig. 6 shows the four sample synthetic rainy images. Each parameter is illustrated in Table 2.

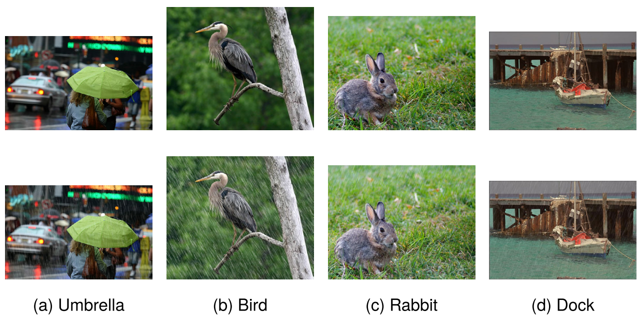

FIGURE 6. Clean images and synthetic rainy images (Rain4).

TABLE 2. Examples of rain streaks parameters.

Image	Size	$\sigma_{T_{\min}}$	σ_l	σ_φ
Umbrella	350×550	0.65	20	100
Bird	480×576	0.66	22	120
Rabbit	640×494	0.57	22	70
Dock	768×512	0.66	22	70

D. DISCUSSION ABOUT COMPOSITE MODELS

As discussed in Section II-C, two main composite models are used when creating rainy images. One is linear additive composite model (2), which is adequate to create heavy rain images and inadequate to images with white part. The other is non-linear screen blend model (3). That model generates natural rainy images but unnatural heavy rain. Although both models have advantages and disadvantages, general state-of-the-arts use only one of them. This hypothesis might not be able to deal with a wide range of real-world images due to the effect of internal reflections [30]. Therefore, we adopt these two models to prepare a rainy image dataset. Supposing several rainy patterns enables our proposed network robust enough to handle a variety of actual rain.

IV. EXPERIMENTAL RESULTS

We evaluate the performance of our de-raining method by conducting experiments on both synthetic rainy images and real-world images. We compare the de-raining performance with three state-of-the-art methods. We select the prior-based method (DSC [6]), the simple CNN-based method (Derain-Net [11]) and the GAN-based method (DID-MDN [13]). All of these methods are implemented with the source codes the authors distribute.

A. TRAINING DATASET

Since real rainy images and the corresponding sunny images are not available simultaneously, we have to create synthetic many rainy images. According to the procedure as discussed in Section III-D, we combine clean images and generated rain streaks to prepare training dataset. Our dataset includes a total of 4900 images including 900 images used in [11] and

TABLE 3. Quantitative comparisons with state-of-the-art using PSNR (dB) on synthetic test images. In the upper part, results of images from Rain4 are shown. The lower part represents an average of each dataset. Red-letter and blue-letter values indicate the highest and the second highest PSNR, respectively.

Images	Rainy image	DSC [6]	DerainNet [11]	DID-MDN [13]	Ours
Umbrella	26.58	31.68	26.30	29.24	35.93
Bird	18.37	23.77	19.23	26.75	27.79
Rabbit	23.89	26.98	19.16	25.04	29.86
Dock	29.58	29.28	29.16	28.42	34.49
Rain12	28.82	28.71	28.96	26.58	31.38
Rain100	21.15	21.04	21.71	21.08	22.11
BSD100	22.48	26.65	22.89	23.78	30.76
Urban100	22.71	26.21	22.71	21.27	30.38

TABLE 4. Quantitative comparisons with state-of-the-art using SSIM on synthetic test images. In the upper part, results of images from Rain4 are shown. The lower part represents an average of each dataset. Red-letter and blue-letter values indicate the highest and the second highest SSIM, respectively.

Images	Rainy image	DSC [6]	DerainNet [11]	DID-MDN [13]	Ours
Umbrella	0.858	0.910	0.902	0.924	0.976
Bird	0.729	0.815	0.847	00.910	0.924
Rabbit	0.942	0.985	0.908	0.914	0.951
Dock	0.941	0.972	0.960	0.923	0.980
Rain12	0.910	0.916	0.939	0.918	0.942
Rain100	0.768	0.764	0.831	0.886	0.812
BSD100	0.841	0.878	0.898	0.854	0.953
Urban100	0.888	0.891	0.888	0.752	0.955

4000 images used in [13]. During a training phase, we randomly crop $384 \times 384 \times 3$ patches. In addition, these patches are randomly flipped horizontally or vertically for data augmentation. Unlike several conventional methods, we create completely random rain noise pattern with different angles, scales and densities. Specifically, we set rain noise parameters $\boldsymbol{\nu}(\sigma_{T_{\min}}, \sigma_l, \sigma_\varphi, \sigma_s, \sigma_a)$ to $\sigma_{T_{\min}} \in [0.54, 0.62]$, $\sigma_l \in [5, 15]$, $\sigma_\varphi \in [50, 130]$, $\sigma_s \in [1.1, 1.5]$,and $\sigma_a \in [0, 2]$. Note that $\sigma \in [\sigma^a, \sigma^b]$ represents σ is taken the value within a range of σ^a to σ^b. A clean image and the rain noise are synthesized by either an additional model or a screen blend model for each patch. We directly pad zeros before convolution to keep an output image same as the input one.

B. LEARNING PARAMETER

We start the training with a base learning rate of 0.0002 and use Adam solver to optimize training parameters. In Adam, we set two learning rates as $\beta_1 = 0.5$ and $\beta_2 = 0.999$. The Pytorch framework is used to train our entire network. The models are trained for up to 4900×40 iterations with a batch size of four. During training we set the adversarial loss ratio in 6 to $\mu_1 = 1$, $\mu_2 = 0.001$.

C. RESULTS ON SYNTHETIC IMAGES

We compare the performance of three previous methods both qualitatively and quantitatively on several synthetic rainy image datasets. We test on commonly used test datasets "Rain12" [11] and "Rain100" [13]. Furthermore, we newly create synthetic rainy images using clean outdoor image datasets from "Rain4 " [11], "BSD100", and "Urban100". In the dataset Rain4, rain noise parameters are set as Table 2.

As a subjective evaluation, Fig. 7 shows the visual comparison for two synthetic rainy images. The second row and the fourth one are the histogram of the first row and the third one respectively. As can be seen in the third column, method [6] leaves a considerable amount of rain streaks. For method [11] in the forth column, rain is almost entirely removed but the overall image turns whitish, specifically feathers of the bird. This color shift is caused by combining low frequency componets with de-rained high frequency components. Method [13] results in over de-raining and losing important details. The histogram shows that the color of their de-rained images are changed. We can see that the peak of the histogram are shifted from the ground truth histogram. In contrast, our proposed model can remove rain streaks well with the contrast of the images remained. The combination of residual learning and UNet structure leads to these great performance.

On the other hand, as an objective evaluation, we compare the de-raining performance using PSNR and SSIM [32] in Table 3 and 4. PSNR results in Table 3 demonstrate that the whiteness throughout an image in method [11] causes a lower PSNR. Also, SSIM results in Table 4 reveal that under de-raining in [6] and over de-raining in [13] lead to lower SSIM. Meanwhile, since our proposed method can remove rain streaks without losing the important details, we achieve higher PSNR and SSIM than other conventional methods.

D. RESULTS ON REAL-WORLD IMAGES

In this section, we make sure whether our proposed method is also applicable to real-world rainy images. We collect many real-world rainy images from the Internet and the test dataset

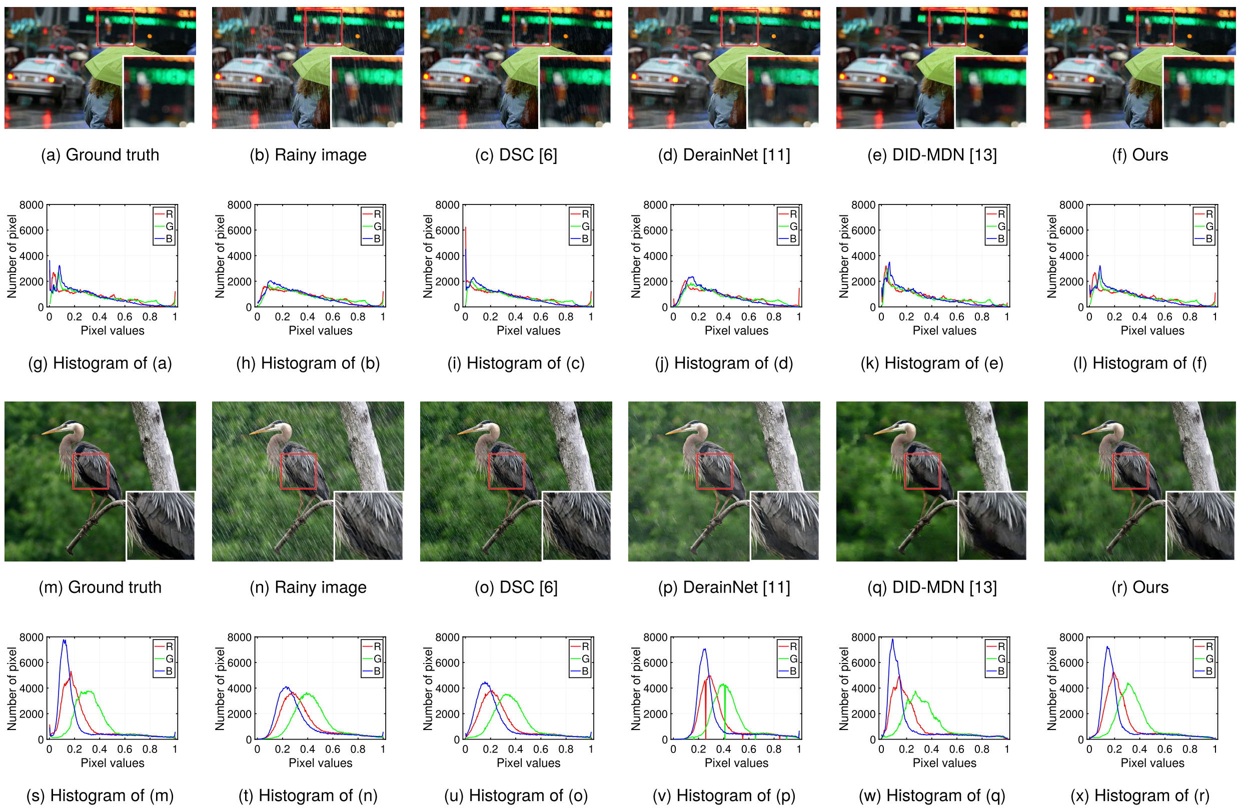

FIGURE 7. Results on two synthetic test images from Rain4, (first row) "Umbrella" and (third row) "Bird". The second row and the forth one are histogram of the first row and the third one respectively.

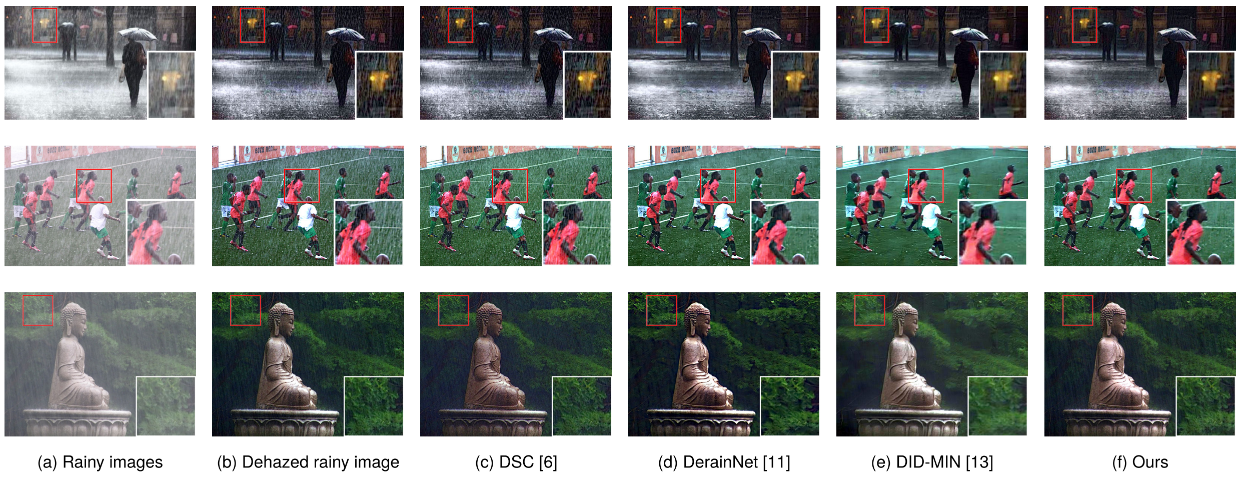

FIGURE 8. Three results on real-world rainy images, "Street" (top), "Soccer" (middle) and "Buddha" (bottom). All algorithms use image de-hazing [31] as a post-processing.

used in [11]. While synthetic images have only rain streaks, real-world images include haze as well as rain streaks. For clear appearance, we apply de-hazing method [31] as a post-processing. We compare visually de-raining performance with state-of-the-art methods. Fig. 8 shows the visual comparison. As can be seen, method [6] suffer from under de-raining for all images. Although method [11] can remove rain streaks, they blur textures and edges. Results of method [13] are over de-rained and they lose important details. In contrast, our proposed method can remove rains with the detail textures preserved. One can see that our model works well not only for synthetic images but also for real-world ones.

E. COMPARISON WITH OTHER COMPOSITE MODELS

As discussed in section III-D, we demonstrate that our proposed composite models are effective for a real-world

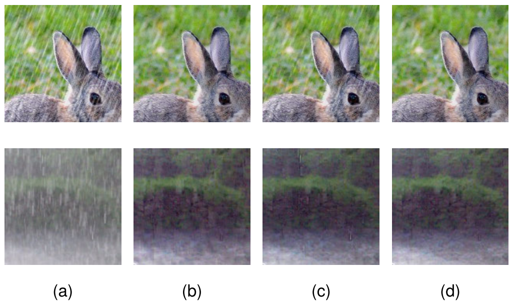

FIGURE 9. Comparison of three different training data. Top: Synthetic image. Bottom: Real-world image. (a) input image, (b) additive composite model, (c) screen blend model and (d) proposed model.

rain removal task. We prepare the following three training datasets. The first one and the second one are synthesized with additive composite model (2) and screen blend model (3), respectively. The third dataset includes synthesized images produced by either of them. As can be seen in Fig. 9, a combination of two composite models is effective for de-raining. Compared with the other models, the network trained with the mixture of two composite models does not leave rain streaks or artifacts very much.

V. CONCLUSION

We have proposed a GAN-based de-raining network trained with mixture of two rain image composite models. To capture global features and local features of rain streaks, we use UNet structure as the generator. In addition, residual learning improves the performance of training and testing process. Moreover, to generate synthetic rain noise for training, we introduce a completely automatic rain noise generator. We compare the de-raining performance with several state-of-the-art methods for synthetic images and real-world images. The results on synthetic data demonstrate that our proposed model noticeably outperforms other conventional methods both quantitatively and qualitatively. For real-world data, our diverse rainy dataset can remove rain streaks and enhance the hazy images without losing important details. Thus, our proposed method overcomes under de-raining, over de-raining and hue change problems.

REFERENCES

[1] T. Matsui, T. Fujisawa, T. Yamaguchi, and M. Ikehara, "Single-image rain removal using residual deep learning," in *Proc. 25th IEEE Int. Conf. Image Process. (ICIP)*, Oct. 2018, pp. 3928–3932.
[2] K. Garg and S. K. Nayar, "Detection and removal of rain from videos," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, vol. 1, Jun./Jul. 2004, pp. I-528–I-535.
[3] V. Santhaseelan and V. K. Asari, "Utilizing local phase information to remove rain from video," *Int. J. Comput. Vis.*, vol. 112, no. 1, pp. 71–89, Sep. 2014.
[4] M. Li, Q. Xie, Q. Zhao, W. Wei, S. Gu, J. Tao, and D. Meng, "Video rain streak removal by multiscale convolutional sparse coding," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 6644–6653.
[5] L.-W. Kang, C.-W. Lin, and Y.-H. Fu, "Automatic Single-Image-Based rain streaks removal via image decomposition," *IEEE Trans. Image Process.*, vol. 21, no. 4, pp. 1742–1755, Apr. 2012.
[6] Y. Luo, Y. Xu, and H. Ji, "Removing rain from a single image via discriminative sparse coding," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Dec. 2015, pp. 3397–3405.
[7] L. Zhu, C.-W. Fu, D. Lischinski, and P.-A. Heng, "Joint bi-layer optimization for single-image rain streak removal," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 2526–2534.
[8] 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, Jan. 2014.
[9] Y.-L. Chen and C.-T. Hsu, "A generalized low-rank appearance model for spatio-temporally correlated rain streaks," in *Proc. IEEE Int. Conf. Comput. Vis.*, Dec. 2013, pp. 1968–1975.
[10] Y. Li, R. T. Tan, X. Guo, J. Lu, and M. S. Brown, "Rain streak removal using layer priors," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 2736–2744.
[11] X. Fu, J. Huang, X. Ding, Y. Liao, and J. Paisley, "Clearing the skies: A deep network architecture for single-image rain removal," *IEEE Trans. Image Process.*, vol. 26, no. 6, pp. 2944–2956, Jun. 2017.
[12] X. Fu, J. Huang, D. Zeng, Y. Huang, X. Ding, and J. Paisley, "Removing rain from single images via a deep detail network," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 3855–3863.
[13] H. Zhang and V. M. Patel, "Density-aware single image de-raining using a multi-stream dense network," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 695–704.
[14] X. Fu, B. Liang, Y. Huang, X. Ding, and J. Paisley, "Lightweight pyramid networks for image deraining," *IEEE Trans. Neural Netw. Learn. Syst.*, to be published.
[15] W. Yang, R. T. Tan, J. Feng, J. Liu, S. Yan, and Z. Guo, "Joint rain detection and removal from a single image with contextualized deep networks," *IEEE Trans. Pattern Anal. Mach. Intell.*, to be published.
[16] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778.
[17] K. Zhang, W. Zuo, Y. Chen, D. Meng, and L. Zhang, "Beyond a Gaussian denoiser: Residual learning of deep CNN for image denoising," *IEEE Trans. Image Process.*, vol. 26, no. 7, pp. 3142–3155, Jul. 2017.
[18] P. C. Barnum, S. Narasimhan, and T. Kanade, "Analysis of rain and snow in frequency space," *Int. J. Comput. Vis.*, vol. 86, nos. 2–3, pp. 256–274, Jan. 2009.
[19] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *Proc. Int. Conf. Med. Image Comput. Comput.-Assist. Intervent.* Springer, 2015, pp. 234–241.
[20] Z. Li, J. Zhang, Z. Fang, B. Huang, X. Jiang, Y. Gao, and J.-N. Hwang, "Single image snow removal via composition generative adversarial networks," *IEEE Access*, vol. 7, pp. 25016–25025, 2019.
[21] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Proc. Adv. Neural Inf. Process. Syst.*, 2014, pp. 2672–2680.
[22] M. Mirza and S. Osindero, "Conditional generative adversarial nets," 2014, *arXiv:1411.1784*. [Online]. Available: http://arxiv.org/abs/1411.1784
[23] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen, "Improved techniques for training GANs," in *Proc. Adv. Neural Inf. Process. Syst.*, 2016, pp. 2234–2242.
[24] A. Creswell and A. A. Bharath, "Task specific adversarial cost function," 2016, *arXiv:1609.08661*. [Online]. Available: http://arxiv.org/abs/1609.08661
[25] J. Zhao, M. Mathieu, and Y. LeCun, "Energy-based generative adversarial network," 2016, *arXiv:1609.03126*. [Online]. Available: http://arxiv.org/abs/1609.03126
[26] S. Reed, Z. Akata, X. Yan, L. Logeswaran, B. Schiele, and H. Lee, "Generative adversarial text to image synthesis," 2016, *arXiv:1605.05396*. [Online]. Available: http://arxiv.org/abs/1605.05396
[27] C. Ledig, L. Theis, F. Huszar, J. Caballero, A. Cunningham, A. Acosta, A. Aitken, A. Tejani, J. Totz, Z. Wang, and W. Shi, "Photo-realistic single image super-resolution using a generative adversarial network," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 4681–4690.
[28] R. A. Yeh, C. Chen, T. Yian Lim, A. G. Schwing, M. Hasegawa-Johnson, and M. N. Do, "Semantic image inpainting with deep generative models," 2016, *arXiv:1607.07539*. [Online]. Available: http://arxiv.org/abs/1607.07539

[29] Z. Zhou, M. Mahfuzur Rahman Siddiquee, N. Tajbakhsh, and J. Liang, "UNet++: A nested U-Net architecture for medical image segmentation," 2018, *arXiv:1807.10165*. [Online]. Available: http://arxiv.org/abs/1807.10165
[30] K. Garg and S. K. Nayar, "Photometric model of a rain drop," Dept. Comput. Sci., Columbia Univ., New York, NY, USA, Tech. Rep., Sep. 2004.
[31] K. He, J. Sun, and X. Tang, "Single image haze removal using dark channel prior," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 33, no. 12, pp. 2341–2353, Dec. 2011.
[32] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: From error visibility to structural similarity," *IEEE Trans. Image Process.*, vol. 13, no. 4, pp. 600–612, Apr. 2004.

TAKURO MATSUI (Student Member, IEEE) received the B.E. degree in electrical engineering from Keio University, Yokohama, Japan, in 2018, where he is currently pursuing the M.E. degree under the supervision of Prof. M. Ikehara. His research interests are in the areas of image denoising and neural networks.

MASAAKI IKEHARA (Senior member, IEEE) received the B.E., M.E., and Dr.Eng. degrees in electrical engineering from Keio University, Yokohama, Japan, in 1984, 1986, and 1989, respectively. He was a Lecturer with Nagasaki University, Nagasaki, Japan, from 1989 to 1992. In 1992, he joined the Faculty of Engineering, Keio University. From 1996 to 1998, he was a Visiting Researcher with the University of Wisconsin–Madison, Madison, and Boston University, Boston, MA, USA. He is currently a Full Professor with the Department of Electronics and Electrical Engineering, Keio University. His research interests are in the areas of multirate signal processing, wavelet image coding, and filter design problems.