Removing Snow from a Single Image Using a Residual Frequency Module and Perceptual RaLSGAN

Thaileang Sung1, Hyo Jong Lee1,2, Member, IEEE
1Department of Computer Science and Engineering, Jeonbuk National University, 567 Baekje-daero, Deokjin-gu, Jeonju-si, Jeollabuk-do, South Korea
2Department of Computer Science and Engineering, CAIIT, Jeonbuk National University, 567 Baekje-daero, Deokjin-gu, Jeonju-si, Jeollabuk-do, South Korea
Corresponding author: Hyo Jong Lee (hlee@chonbuk.ac.kr).

ABSTRACT Removing snow particles from an image is a complicated task due to the shape, size, and color of the particles. The latest snow removal methods remove snow from a single image but still retain some snow and salt-and-pepper particles. Some approaches, while trying to remove snow from a single image, produce blurry artifacts. In this paper, we solve these problems by designing a network model that consists of a residual generative network, a snow-free image generative network, and a perceptual relativistic discriminative network. In both generative networks, we assign the residual frequency network (ReFNet) as our bottleneck module. Our network model learns to map two relationships. First, the input snowy image is trained to map the snow mask image in the dataset. Then, a retained image resulting from subtraction between an input image and the estimated residual image is concatenated with the input snowy image and mapped to the desired snow-free ground truth. Moreover, we use a perceptual identical-paired adversarial network based on a relativistic discriminative network to make our training results more robust. Our results achieve greater performance than state-of-the-art methods on both synthetic and real-world snowy images.

INDEX TERMS Snow removal, de-snow, deep learning, residual network, convolutional neural network, discrete wavelet transformation, relativistic discriminator, perceptual adversarial network.

I. INTRODUCTION

Bad weather influences outdoor vision systems, such as security or surveillance cameras, auto-driving systems, and traffic monitors. Its influence can cause misjudgment by humans or vision analytic applications, such as object detection, segmentation, and action recognition. In general, these applications require sharp and clean images to handle their related tasks and produce correct outputs. However, in a real situation, the acquired images are not always clean due to bad weather interferences such as haze, fog, rain, and snow. Therefore, image enhancement and restoration techniques must be investigated.

Numerous approaches have been proposed to try and remove atmospheric particles from images. Early approaches tried to deal with haze particles, such as [1-6], which used prior-based methods. However, those methods did not work well in certain cases due to the inaccurate estimation of a transmission map, haze-relevant prior, or heuristic cues. During the rise of deep learning, many deep learning-based methods were also proposed, including DehazeNet [7], residual learning methods [8], densely connected pyramid dehazing networks [9], and FFA-Net [10]. These approaches approximated the dehazing’s purpose. Furthermore, another essential challenge is rain removal; while haze is uniformly accumulated over an entire image, rain has non-uniform densities, diverse shapes, streaks, and various orientations. Like dehazing, its solutions started with traditional methods. Some tried to deal with a certain type of rainy scene [11] regardless of the shape, size, and density of the rain streaks, which caused issues like under de-raining, over de-raining, and over-smoothing of the artifacts. Deep learning-based methods, such as [12-18], obtained greater performance than previous traditional methods. Fu et al. [12] and Yang et al. [17] built a large synthetic rain dataset consisting of various levels
of rain density, shapes, and orientations for training with learning-based methods. Though some of the learning-based methods mentioned above can be considered as generic methods for removing atmospheric particles, Liu et al. [19] claimed that it is difficult to adapt those methods to snow removal due to the complications of snow characteristics, including uneven density, diverse particle shapes and sizes, transparency, and irregular trajectory. They introduced a synthetic Snow100K dataset consisting of 100K synthesized snowy images, corresponding snow-free ground truths, snow masks, and an additional 1,329 real-world snowy images. They produced 5.8K base masks via Photoshop to synthesize the snowy images. With this dataset, they also designed a multistage network, named DesnowNet, for translucent and opaque snow particle removal. For more accurate estimation, they differentiated snow into attributes of translucency and chromatic aberration. Snow has complicated characteristics, and manually synthesizing snow via Photoshop cannot produce all types of snow characteristics. Consequently, DesnowNet obtained effective evaluation results on synthetic test images, but it failed to produce satisfactory snow-free images from real-world snowy images. Li et al. [20] used the same Snow100K dataset to train their model. They proposed composition generative adversarial networks (CGANs) to remove snowflakes from a single image with different sizes of snow particles. They stated that the mean absolute error (MAE) function generates an image that is close to ground truth faster as it deals with only the pixel domain. Though CGAN outperformed DesnowNet on both synthetic and real-world snowy images, it still led to blurry artifacts in the resultant images. Isola et al. [21] claimed that using L1 in their objective loss function improved the training speed, but they also produced blurry reconstructed images.

In this paper, we design a snow removal model that can preserve details and avoid blurry artifacts after image reconstruction, regardless of the diverse snow characteristics (i.e., orientation, shape, and translucency). The proposed model consists of three networks: a residual generator network, a snow-free generator network, and a perceptual discriminator network. Applying these networks, we propose a bottleneck module, which we refer to as the residual frequency network (ReFNet). ReFNet consists of a discrete wavelet transform (DWT) [22-24] and CNNs. The network has a similar structure as Res2Net [25], but the number of its scales is fixed to four; in the module, each scale represents a feature-frequency component. Based on DWT, there are four frequency components, i.e., one low-level frequency and three high-level frequencies. The conventional discrete wavelet transform [22, 26] removes salt-and-pepper noise effectively. Liu et al. [14] realized the advantages of DWT and proposed a multi-level wavelet-CNN for image restoration. Their method removes not only salt-and-pepper noise, but also preserves detailed information of the restoration images. Therefore, we adapt DWT into our bottleneck module. It is, structurally, constructed as a pooling layer [24] in ReFNet. It serves our purpose of removing snow particles with various characteristics, such as a salt-like shape, which were disregarded in existing data-driven methods. Furthermore, it can preserve details after its reconstruction. ReFNet helps maintain high frequency areas or objects of a reconstructed image that are mapped to a ground truth, while the whole network considers whether snowflakes belong to the high-frequency domain or the snowflakes themselves are high frequency. We also modified RaLSGAN [27] into perceptual RaLSGAN. We adapt the iPANs discriminator’s logic [28] and perceptual adversarial network [29] to build our adversarial loss functions for the perceptual RaLSGAN. In addition, we add the mixed loss [30] and composition loss to the generative adversarial loss function to improve the robustness of our final results.

Our approach has contributed to the realm of image enhancement and restoration as follows:

- We propose a novel network architecture that produces an estimated snow mask and snow-free image, outperforming state-of-the-art processes on both synthetic and real-world snowy images.  
- We also propose ReFNet as our bottleneck module. It is applied in both the residual generator network and snow-free image generator network. ReFNet enables our model networks to remove snow of different characteristics and preserve the delicate parts of images.  
- We update the existing relativistic discriminator network into a perceptual relativistic discriminator network to improve the robustness of our training results.

We organize the rest of this paper by sections, as follows. In Section II, we describe related works that are relevant to our approach. We explore our proposed methods in Section III, including the network architecture and equations. Section IV describes our training environment and parameters setting. We
exhibit our quality and visual results in Section V and conclude our paper in Section VI.

II. RELATED WORKS

Below, we review related snow removal methods, discrete wavelet transforms, and the relativistic discriminator.

A. Snow Removal

Single-image snow removal is a technique used to remove snowflakes or restore a clear and clean image from a snowy image. Pei et al. [31] assumed that the snowflakes are located in the high-frequency and bright areas. They analyzed the color and frequency features of snowy images, extracted the high-frequency part, and detected snowflakes using intensity prior. However, using intensity characteristics led to the misidentification of highlighted regions of the background as snowflakes. Zheng et al. [32] used the same assumption. The high-frequency region is decomposed into background edges and snowflakes using a guided filter, and low-frequency components are used for the guide image. This method followed rain characteristics and disregarded the transparency, scale, shape, trajectory, and distribution of snow characteristics. To get away from rainfall-driven feature approaches, Liu et al. [19] proposed DesnowNet, which is based on a multistage network, to remove translucent and opaque snow particles. As mentioned in the introduction, their great contribution to snow removal is that they created the Snow100K² dataset, which consists of diverse snow characteristics. However, 5.8k snow base masks are insufficient for the snow characteristics. Therefore, DesnowNet still produces an unclear image from a real-world snowy image. Li et al. [20] proposed CGAN in order to generate a cleaner image. They trained their network on the same Snow100K² dataset and used an up-to-date loss function based on least squares error. They used the mean absolute error loss function, which is sensitive to outliers. Therefore, the snow particles are well removed; however, as mentioned in [28, 33], even though the loss function can make the predicted image become closer to its ground truth, it leads to a blurry effect.

B. Discrete Wavelet Transform and Wavelet-CNN

The discrete wavelet transform (DWT) [22] can decompose, de-noise, and analyze vibration signals. It decomposes low-frequency and high-frequency pass filters. DWT is a well-known preserving method. Wavelet-SRNNet [34] predicts a series of corresponding wavelet coefficients and design losses in the wavelet domain to capture global topology information and local textural details. Liu et al. [14] proposed a multi-level wavelet network (MWCN) based on the U-Net [35] architecture. DWT is built inside the contracting subnetwork of MWCNN to replace the pooling operation and guarantee that all of an image’s information can be preserved. Through this, we create a residual frequency module based on the Haar wavelet transform, which is capable of learning high-frequency components and distinguishing between the frequency of the background and snow particles. More importantly, the detailed texture of our generated image can be well preserved.

C. Relativistic Discriminator

The relativistic discriminator [27] was proposed to fix and improve the standard discriminator [36, 37], which had missing properties related to the prior knowledge, divergence minization, and gradient. While the standard GAN has discriminator \( D(x) \), where \( x \) is either real or fake, the relativistic discriminator depends on both real and fake data \( \tilde{x} = (x_r, x_f) \); the relativistic discriminator is \( D(\tilde{x}) \). See Equations (13) and (14) in Section III. It is proven that the relativistic discriminator is generally stable and improves data quality. There are many relativistic discriminator preferences proposed in [27], i.e., relativistic standard GAN (RSGAN), relativistic average standard GAN (RaSGAN), and relativistic least square GAN (RaLSGAN). However, RaLSGAN provides the best results and is more stable than the others. According to iPANs [28], perceptual adversarial loss produces better results compared to vanilla adversarial loss. Thus, in this study, RaLSGAN is extended into perceptual-RaLSGAN (pRaLSGAN) with respect to the iPANs discriminator, which is discussed in Section III.D in detail.

III. PROPOSED METHOD

We now illustrate our proposed method to remove snow from a snowy image. We describe our network architecture and explain how DWT and its inversion are applied to input feature maps. We also derive our methodological equations, such as residual loss \( L_2 \), composition loss, mix-loss, and iPANs-based pRaLSGAN loss functions.

A. Network Architecture

We show our network architecture in Fig. 2, which consists of three parts, i.e., a residual image generator, a snow-free-image generator, and a discriminator.

The residual image generator is a network used for acquiring a binary snow image. First, RGB snowy image \( x \) with the size \( 64 \times 64 \) is assigned into the network. The initial layer is assisted by convolution, batch normalization (BN), and the LeakyReLU activation function. Then, DWT transforms the output feature map into four components of frequency feature maps, i.e., low-level frequency \( L_L \) and high-level frequencies \( H_L, L_H, \) and \( H_H \). The size of each frequency feature is \( 64 \times 32 \times 32 \). We discuss DWT further in the next section (Section III.B). Each component is sent through five ReFNNets (frequency blocks) and then the feature outputs are concatenated. Next, the joint wavelet transformation features are inverted and input into the final hidden layer, which has convolution, BN, and PReLU components.

The generated residual image \( r \) is subtracted from the input
We also express our residual generator mathematically in Equation (1). Suppose that an input snow image $x \in [0, 1]^{3 \times m \times n}$ with size $m \times n$ is also composed of a $y \in [0, 1]^{3 \times m \times n}$ snow-free image and an $r \in [0, 1]^{3 \times m \times n}$ residual snow image.

$$z_1 = a(BN(w_1 \odot x)).$$
$$dwt(z1) \rightarrow [LL, HL, LH, HH],$$
$$F = \left\{ f_{LL}, f_{HL}, \ldots, f_{IH}(LL) \ldots \right\},$$
$$f_{LL}, f_{HL}, \ldots, f_{IH}(HL) \ldots, f_{IH}, f_{HL} \ldots (f_{IH}(HH) \ldots)$$
$$iwt(F) \rightarrow iwt(c(f_{LL}, f_{HL}, f_{IH}, f_{HH})).$$
$$z_2 = p(BN(w_2 \odot iwt(F)).$$
$$r = p(BN(w_1 \odot iwt(F(dwt(z_{l-1}))))).$$

(1)

Here, $l$ is the number of layers and the symbol $\odot$ indicates the convolution operator between the weight $w$ and the input features or image $x$. The variables $a$ and $p$ represent LeakyReLU and PReLU, respectively, considering the activation function at the last layer of the residual image generator. $BN(\cdot)$ is the batch normalization and $dwt(\cdot)$ is a DWT pooling layer where its inversion is $iwt(\cdot)$. $F$ represents the residual frequency blocks. After transforming through the dwt layer and splitting the input feature map into four different frequency features i.e., $LL, HL, LH$, and $HH$, each one of these frequency features is input into residual frequency block $f$, expressed as $f_{LL}, f_{HL}, f_{IH}, f_{HH}$. In this generator, we use five $f$ frequency modules (ReFNets) for each frequency stream, e.g. $f_{LL}, f_{HL}, \ldots, (f_{IH}(LL) \ldots)$. The ReFNets module is described in Section III.C in detail. Then $c$ represents a concatenation of all the frequency outputs before the inversion of the wavelet transformation.

Subtracting a residual snow image from a snowy image does not produce the desired snow-free image, but leads to a black-snow image, as shown in Fig. 3. Logically, there is nothing but the default black color behind each falling snowflake of the input image $x$.

where $c(x, (x - r))$ is concatenated with the snowy input image and input into another generative network: the snow-free generator. The advantage of concatenating white snow and black snow here allows our network to learn about synthetic snow in the dataset, as well as a type of pepper-like snow, which has been disregarded in existing works. The concatenated image has six channels and we acquire 64

The image obtained from $(x - r)$ is concatenated with the snowy input image and input into another generative network: the snow-free generator. The advantage of concatenating white snow and black snow here allows our network to learn about synthetic snow in the dataset, as well as a type of pepper-like snow, which has been disregarded in existing works. The concatenated image has six channels and we acquire 64

![FIGURE 3. Black-snow image of a real-world snowy image resulting from subtracting the residual image.](image-url)
channels at a size of $64 \times 64$ in the first layer of the snow-free generative network. Like the residual image generator, we assign the output feature maps through 16 blocks of ReFNet. So, except for the number of ReFNet, we use the same size and layers as used in the residual image generator. We express the free-snow image generator as shown in Equation (2) below:

$$
\begin{align*}
    z_1 &= a\left( BN(w_1 \odot c((x - r), x)) \right), \\
    F &= f_1(\ldots(f_{16}(z_1)) \ldots), \\
    z_2 &= p(BN(w_2 \odot F), \\
    &\rightarrow y = p \left( BN(w_t \odot \ldots(f_{16}(z_{t-1})) \ldots) \right).
\end{align*}
$$

(2)

B. Discrete Wavelet Transform Layer

We borrow the idea of how to decompose an input $x$ into four sub-band frequency features, such as $LL$, $HL$, $LH$, and $HH$, from [23]. This is done by using 2D DWT with four convolutional filters, i.e., the low-pass filter $f_{LL}$ and the high-pass filters $f_{HL}, f_{LH},$ and $f_{HH}$, based on the Haar wavelet transformation technique. The operation of DWT is defined as:

$$
\begin{align*}
    LL &= (f_{LL} \circ x) \downarrow 2, \\
    HL &= (f_{HL} \circ x) \downarrow 2, \\
    LH &= (f_{LH} \circ x) \downarrow 2, \\
    HH &= (f_{HH} \circ x) \downarrow 2.
\end{align*}
$$

(3)

where $\downarrow 2$ represents the standard down-sampling operator with a factor of 2. We can rewrite Equation (3) according to the Haar transform theory [38] with the $(i,j)th$ values of $LL$, $HL$, $LH$, and $HH$ as

$$
\begin{align*}
    LL(i,j) &= x(2i - 1, 2j - 1) + x(2i - 1, 2j) + x(2i, 2j - 1) + x(2i, 2j), \\
    HL(i,j) &= -x(2i - 1, 2j - 1) + x(2i - 1, 2j) - x(2i, 2j - 1) + x(2i, 2j), \\
    LH(i,j) &= -x(2i - 1, 2j - 1) - x(2i - 1, 2j) + x(2i, 2j - 1) + x(2i, 2j), \\
    HH(i,j) &= x(2i - 1, 2j - 1) - x(2i - 1, 2j) - x(2i, 2j - 1) + x(2i, 2j).
\end{align*}
$$

(4)

In our network architecture, we use DWT once in the residual generative network and once in ReFNet. Each DWT is reconstructed by inverse wavelet transformation (IWT). For example, DWT in the residual generative network is reconstructed after getting results from the ReFNet blocks; however, all outputs must be concatenated first. Following the work in [23], IWT can be written as follows:

$$
\begin{align*}
    x(2i - 1, 2j - 1) &= (LL(i,j) - LH(i,j) - HL(i,j)) + HH(i,j))/4, \\
    x(2i - 1, 2j) &= (LL(i,j) - LH(i,j) + HL(i,j) - HH(i,j))/4, \\
    x(2i, 2j - 1) &= (LL(i,j) + LH(i,j) - HL(i,j)) - HH(i,j))/4.
\end{align*}
$$

We use the same DWT and IWT technique in ReFNet.

C. Residual Frequency Network

We use ReFNet as our bottleneck network. As shown in Fig. 2 and Fig. 4, ReFNet starts with a convolution, BN, and LeakyReLU at the size of $64 \times 32 \times 32$. Similar to the multi-scale features approach of Res2Net [25], we have four features scales. However, our features have been defined in specific formats using the DWT pooling layer explained in the discrete wavelet transformation layer section, such as low-pass $LL$ and high-pass $HL, LH,$ and $HH$.

Different from Res2net, we don’t divide the input and output channels by the size of the scale in the conv3x3 and BN because the four features are independent after being decomposed and the size of the output is intentionally reconstructed to be the same as the input feature’s size after using IWT. We express ReFNet with the following equations:

$$
\begin{align*}
    z_1 &= a(BN(w_1 \odot x)), \\
    dwt(z_1) &\rightarrow [LL, HL, LH, HH], \\
    f_{LL} &= LL, \\
    f_{HL} &= a(BN(w_{hl} \odot HL)), \\
    f_{LH} &= a(BN(w_{lh} \odot LH)) + f_{HL}, \\
    f_{HH} &= a(BN(w_{hh} \odot HL)) + f_{HL}, \\
    F_1 &= c(f_{LL}, f_{HL}, f_{LH}, f_{HH}), \\
    iwt(F_1) &\rightarrow iwt(c(f_{LL}, f_{HL}, f_{LH}, f_{HH})), \\
    z_2 &= a(BN(w_2 \odot iwt(F_1)), \\
    &\rightarrow g_{ReFNet} = a(BN(w_t \odot iwt(F_{t-1}))).
\end{align*}
$$

(6)

In Equation (6), the feature $x$ is input into the first layer of conv1x1, BN, and LeakyReLU. Then, DWT decomposes the first feature map to $LL, HL, LH$, and $HH$. While $LL$ is maintained $HL, LH,$ and $HH$ are trained with conv3x3, BN, and LeakyReLU, as shown in Equation (6) and Fig. 4. Feature $f_{HL}$ is added to a convolutional layer of feature $f_{LH}$, and feature $f_{LH}$ is also added to a convolutional layer of feature $f_{HH}$. All the feature outputs are concatenated and reconstructed with IWT. Lastly, the inverted output is assigned through conv1x1, BN, and LeakyReLU, and then feature $g_{ReFNet}$ is obtained as the final output of ReFNet.
D. Discriminative Network

There are two major focuses in our discriminator networks: the structure of the networks and the formation of loss functions.

First, we use the iPANs-based discriminator networks proposed by Sung et al. [28], which use duplicate ground truths as paired inputs for real discrimination and a generated image with the ground truth (desired output) for fake discrimination. iPANs has adapted a perceptual linear network [39] into its discriminator network and jointly modified PANS (perceptual adversarial networks) [29] to find its desired perceptual adversarial loss. PANS calculate perceptual loss by directly subtracting layers of a network from other layers of another network using L2. Additionally, Zhang et al. [39] use a perceptual linear network to calculate the perceptual loss linearly. Thus, iPANs replaces the direct features subtraction loss function is used:

\[ \text{similarity (VOLUME XX, 2017 9) } \]

In Equation (1) of the residual image generator, we obtain \( g_R \), which represents an estimated residual image. Here, the L2 loss function is a combination of multi-scale structural similarity (ms-ssim) and \( L_1 \) loss functions:

\[
\mathcal{L} = \frac{m}{r} \text{ms-ssim} \left( y, y_{gt} \right) + \left( 1 - \frac{m}{r} \right) L_1,
\]

where \( m \) is the total number of input data, \( r \) is iterated from 1, 2, ..., \( n \), and \( m \) is the snow mask ground truth from the dataset.

In Equation (2), we obtain \( y \) as our desired snow-free image. To refine our result, we use multiple loss functions, such as mix-loss, composition loss, and perceptual adversarial loss.

Mix-loss was originally proposed by Zhao et al. [30]. This loss function is a combination of multi-scale structural similarity (ms-ssim) and \( L_1 \) loss functions:

\[
\mathcal{L}_{\text{mix}} = \alpha \text{ms-ssim} + (1 - \alpha) L_1,
\]

where \( \alpha = 0.84 \) is empirically suggested by the author of [30]. Ms-ssim is assigned as the loss function, as shown below:

\[
\text{ms-ssim} \left( y, y_{gt} \right) = \left( I_{m} \left( y, y_{gt} \right) \right)^{\alpha_m} \cdot \prod_{j=1}^{M} \left( c_{j} \left( y, y_{gt} \right) \right)^{\beta_j} \cdot \left[ \sum_{j=1}^{S} \left( s_{j} \left( y, y_{gt} \right) \right)^{\gamma_j} \right].
\]

E. Loss Functions

In Equation (1) of the residual image generator, we obtain \( r \), which represents an estimated residual image. Here, the L2 loss function is used:

\[
\mathcal{L}_r = \frac{1}{n} \sum_{i=1}^{n} \left\| m_i - r_i \right\|^2, \tag{7}
\]

where \( n \) is the total number of input data, \( i \) is iterated from 1, 2, ..., \( n \), and \( m \) is the snow mask ground truth from the dataset.

In Equation (2), we obtain \( y \) as our desired snow-free image. To refine our result, we use multiple loss functions, such as mix-loss, composition loss, and perceptual adversarial loss.

Mix-loss was originally proposed by Zhao et al. [30]. This loss function is a combination of multi-scale structural similarity (ms-ssim) and \( L_1 \) loss functions:

\[
\mathcal{L}_{\text{mix}} = \alpha \text{ms-ssim} + (1 - \alpha) L_1, \tag{8}
\]

where \( \alpha = 0.84 \) is empirically suggested by the author of [30]. Ms-ssim is assigned as the loss function, as shown below:

\[
\text{ms-ssim} \left( y, y_{gt} \right) = \left( I_{m} \left( y, y_{gt} \right) \right)^{\alpha_m} \cdot \prod_{j=1}^{M} \left( c_{j} \left( y, y_{gt} \right) \right)^{\beta_j} \cdot \left[ \sum_{j=1}^{S} \left( s_{j} \left( y, y_{gt} \right) \right)^{\gamma_j} \right]. \tag{9}
\]

\[
\text{ms-ssim} \left( y, y_{gt} \right) = \left( I_{m} \left( y, y_{gt} \right) \right)^{\alpha_m} \cdot \prod_{j=1}^{M} \left( c_{j} \left( y, y_{gt} \right) \right)^{\beta_j} \cdot \left[ \sum_{j=1}^{S} \left( s_{j} \left( y, y_{gt} \right) \right)^{\gamma_j} \right]. \tag{10}
\]

Three similarity functions, i.e., \( I_m \) luminance, \( c_j \) contrast, and \( s_j \) structure components are combined at different scales. The computation is done in a sliding \( N \times N \) Gaussian-weighted window. In this equation, we use \( N = 11 \), meaning \( 11 \times 11 \). The luminance component \( I_m \) is calculated only at the highest scale \( M \). For our convenience, we set \( \alpha_j = \beta_j = \gamma_j = 1 \). Refer to [40] for more details about multi-scale structural similarity.

The last member of the \( \mathcal{L}_{\text{mix}} \) function is \( L_1 \), which is expressed in Equation (11) below.

\[
L_1 = \frac{1}{n} \sum_{i=1}^{n} \left| y_{gt} - y_i \right|_1 \tag{11}
\]

The composition loss function using \( L_1 \) is part of our total loss function. This is to guarantee that the estimated snow-free image does not encounter any information loss issues caused by artifacts of the estimated residual image at the pixel level.

\[
L_c = \frac{1}{n} \sum_{i=1}^{n} \left| x_i - r_i - y_i \right|_1 \tag{12}
\]

Furthermore, we use a discriminative network where we can find adversarial loss functions. First, we must find generative and discriminative losses depending on our network architecture.

We use the relativistic average LSGAN loss function based on [27], which was originally written as:

\[
\begin{align*}
\mathcal{L}_D &= E_{y_{gt} \sim P} \left[ \left( C(y_{gt}) - E_{y \sim Q} C(y) - 1 \right)^2 \right] + \\
\mathcal{L}_C &= E_{y \sim Q} \left[ \left( C(y) - E_{y_{gt} \sim P} C(y_{gt}) + 1 \right)^2 \right].
\end{align*}
\]

Equations (13) and (14) are RaGAN loss functions, where \( C(\cdot) \) is the non-transformed discriminator output. \( y \) and \( y_{gt} \) are the estimated snow-free images and snow-free ground truth.

Our network architecture uses an identical-paired discriminator, which has two network streams, concatenated at the last layers. Now, we obtain non-transformed discriminator outputs as

\[
\begin{align*}
C(\text{real}) &= C(y_{gt}, y_{gt}) \tag{15} \\
C(\text{fake}) &= C(y, y_{gt}) \tag{16}
\end{align*}
\]

We can then rewrite our equations as follows:
\[
L_D = E_{y_{gt} \rightarrow p} \left[ \left( C(\text{real}) - E_{y} \cdot C(\text{fake}) - 1 \right)^2 + E_{y} \left( C(\text{fake}) - E_{y_{gt} \rightarrow p} C(\text{real}) + 1 \right)^2 \right]. \tag{17}
\]

\[
L_G = E_{y} \left[ \left( C(\text{fake}) - E_{y_{gt} \rightarrow p} C(\text{real}) - 1 \right)^2 + E_{y_{gt} \rightarrow p} \left( C(\text{real}) - E_{y} \cdot C(\text{fake}) + 1 \right)^2 \right]. \tag{18}
\]

Our goal for the discriminator is to create a perceptual adversarial loss. However, we first need to find the perceptual loss. According to our disentanglement network and [28],

\[
L_p = \sum_{j \in I} \frac{1}{H \times W} \sum_{h, w} \left\| w_j \otimes (A_{jhw} - B_{jhw}) \right\|_2^2. \tag{19}
\]

The discriminator has two network streams A and B where \(A_{jhw}, B_{jhw} \in \mathbb{R}^{H \times W \times c_j}\). We calculate the feature distance by extracting feature stacks from \(j\) layers and then unit-normalize their channel’s dimensions. We scale the activation channel-wise \(\otimes\) by vector weight \(w_j \in \mathbb{R}^c\), and then \(L_2\) is computed.

Then, given a positive margin \(m\), we can define the perceptual loss for the discriminator as

\[
L_{p_d} = \left[ m - L_p \right]^+, \tag{20}
\]

where \([.]^+ = \max(0,)\). For more information related to this equation, refer to [29].

From Equations (17) to (20), we obtain pRaLSGAN loss functions:

\[
L_{PG} = \lambda_G L_G + \lambda_p L_p, \tag{21}
\]

\[
L_{PD} = \lambda_D L_D + \lambda_p [m - L_p]^+, \tag{22}
\]

where \(\lambda_G\), \(\lambda_p\), \(\lambda_D\), and \(\lambda_{p_d}\) are hyper-parameters balancing between the loss functions.

Finally, our total objective loss function for snow-free image estimation can be expressed as follows:

\[
L_{total} = \lambda_r L_r + \lambda_{mix} L_{mix} + \lambda_c L_c + \lambda_{PG} L_{PG}, \tag{23}
\]

\[
L_{PD} = \lambda_D L_D + \lambda_p [m - L_p]^+, \tag{24}
\]

and \(L_{PG}\) is an adversarial against \(L_{PD}\) during training.

IV. EXPERIMENTAL SETUP

A. Environment

Our network is trained on a Titan X (Pascal) GPU, where we implemented our method using Pytorch 1.2.0 and Cuda 10.0 in a Python 3.6 environment. We resize the training images to 256×256 and train on randomly cropped 64×64 patches. We set the batch size and learning rate to 32 and 2-4e. We use the Adam optimizer and set \(\beta_1\) and \(\beta_2\) to 0.5 and 0.9, respectively. We define the learning rate to start decay after 50 epochs. Then, we train our network with 200 epochs.

We use the Snow100K\(^2\) dataset, which is the same as the one used in [19]. The dataset consists of 100K synthesized snowy images and corresponding snow-free images, as well as snow masks. The dataset also contains 1,329 realistic snowy images. There are three subsets, which are divided into small, medium, and large snow subsets. Table I shows more detail about the number of training and test sets in the dataset.

B. Experimental Results

Fig. 5 exhibits the visual results from DesnowNet and CGAN, along with our results in the right-most column. Visually, in the top row, we can see that DesnowNet removed distant lights, mistaking them for snowflakes. On the other hand, CGAN deformed the lights and made them look blurry, while our method could maintain the lights’ information and is free from the blurry effect. For the second images in the middle row, DesnowNet couldn’t remove snow well from the region of the red box at the bottom and it produced blurry numbers and letters in the region of the red box at the top. CGAN, even though it can remove more snow than DesnowNet, still yields blurry artifacts. The image produced by DesnowNet in the bottom row contains many flakes of snow that don’t look clean. CGAN looks cleaner, but the image is blurred. As shown in the figure, our method 1) has a cleaner image (snow-free image), 2) shows no blur effect, and 3) avoids information loss. We adopt two commonly used metrics, i.e., PSNR and SSIM, to evaluate our snow removal results quantitatively. Using the Snow100K\(^2\) test dataset, we evaluate the difference between the estimated snow-free image \(y\) and ground-truth \(y_{gt}\). We also looked at the difference between the generated mask \(r\) and ground truth mask.

TABLE II displays the comparison of our approach with existing state-of-the-art methods on the synthetic test dataset. It shows the qualitative comparison results under three test subsets, i.e., small, medium, and large falling snow conditions. We also add an overall column to display the average results over the entire test dataset. DeepLab [41], JORDER [17], DesnowNet [19], and CGAN (composition GAN) [20] are...
chosen for the comparison. Overall, we obtain greater performance compared to other existing works. The SSIM of the proposed method is the best for each test, while CGAN and DesnowNet show slightly better results than the proposed method in terms of the PSNR.

There is a big performance gap between DesnowNet and DeepLab and JORDER. CGAN outperforms DesnowNet in terms of the overall results, as shown in TABLE II, but their results under each subset are competitive; these two approaches have their distinct pros and cons. For instance, DesnowNet reconstructed an output with fewer artifacts than CGAN, but its output remains snowy. Alternatively, CGAN can remove snow better than DesnowNet, but it produces blurry artifacts. Our approach solves both of these existing problems. As a result, we obtain higher performance with fewer artifacts compared to DesnowNet and CGAN.

Furthermore, we also compare our estimated snow mask $r$ in TABLE III with the existing works. CGAN is excluded from this comparison due to the lack of resources and source code provided by the authors. The result of our residual generative network yields the greatest SSIM value (0.646) but has a lower PSNR than DesnowNet (20.536 vs. 22.005).

We show our sample results for the estimated snow-free image and estimated snow mask image from the real-world snowy image input in Fig. 1; our method can handle large and heavy snow, as well as small and medium particle snow.
C. Ablation Study

In this section, we investigate the effect of using different modules in our network architecture. The proposed ReFNet module is motivated by the Res2Net [25] module, as mentioned in Section III.C. Therefore, we train our network architecture with Res2Net and compare its results with the proposed ReFNet architecture. To be fair, we set the scale dimension of Res2Net to $s = 4$, as we also have four frequency components after DWT decomposition. Furthermore, the first and last layers are kept the same for both modules. The network is trained with 200 epochs.

Fig. 6 depicts the visual results of the two modules. At first glance, Res2Net seems to produce clearer results than ReFNet, but if we look closer, we can see that there is remaining snow and noise, as well as color changes in the estimated images. On the other hand, ReFNet maintains the image’s color and removes snow and noise better than Res2Net. ReFNet helps our network produce clearer results and less snow than Res2Net.

### TABLE III

PERFORMANCES OF ESTIMATED MASK ON SNOW100K TEST SET

| Metric     | PSNR | SSIM |
|------------|------|------|
| DeepLab    | 18.867 | 0.294 |
| JORDER     | 19.954 | 0.391 |
| DesnowNet  | 22.005 | 0.566 |
| Ours       | 20.536 | 0.646 |

FIGURE 6. Real-world snowy images and results of Res2Net and ReFNet modules.

FIGURE 7. Comparison of SSIM values between our proposed method with Res2Net and ReFNet.

FIGURE 8. Comparison of PSNR values between our proposed method with Res2Net and ReFNet.
Fig. 7 and Fig. 8 show the comparison results for the proposed method trained with the Res2Net and RefNet modules using SSIM and PSNR metrics. The figures show that RefNet has superior performance compared to Res2Net in almost every epoch, based on the SSIM metric in Fig. 7. However, the results are very competitive when using the PSNR metric, as shown in Fig. 8.

According to the comparison results above, our method led to greater performance than the latest state-of-the-art methods on both synthetic and real-world snowy images.

V. CONCLUSION

In this paper, we proposed a novel network architecture assisted by two generators, including a residual generator and a snow-free generator, and an iPANs-based relativistic discriminator. Each generator has the same bottleneck module: the proposed RefNet. Perceptual RaLSGAN improves our robustness and balances our training.

Our results suggest that our network is superior to current data-driven, state-of-the-art methods. While existing works depend strongly on the accuracy of the dataset, the proposed module uses a DWT layer embedded by CNNs, which allows our method to remove unknown type of snow which is not found in the synthetic dataset. Moreover, DWT and IWT also represent down- and up-sampling layers. They prevent outputs from losing information during down-sampling and up-sampling. Finally, the proposed pRaLSGAN also helps avoid blurry artifacts.

ACKNOWLEDGMENT

The authors would like to acknowledge the financial support received from the Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education (GR 2019R1D1A3A03103736)

REFERENCES

[1] D. Berman and S. Avidan, "Non-local image dehazing," presented at the Proceedings of the IEEE conference on computer vision and pattern recognition.
[2] R. Fattal, "Single image dehazing," ACM transactions on graphics (TOG), vol. 27, no. 3, pp. 1-9, 2008.
[3] R. Fattal, "Dehazing using color-lines," ACM transactions on graphics (TOG), vol. 34, no. 1, pp. 1-14, 2014.
[4] Y. Jiang, C. Sun, Y. Zhao, and L. Yang, "Image dehazing using adaptive bi-channel priors on superpixels," Computer Vision and Image Understanding, vol. 165, pp. 17-32, 2017.
[5] R. T. Tan, "Visibility in bad weather from a single image," presented at the 2008 IEEE Conference on Computer Vision and Pattern Recognition.
[6] Q. Zhu, J. Mai, and L. Shao, "A fast single image haze removal algorithm using color attenuation prior," IEEE transactions on image processing, vol. 24, no. 11, pp. 3522-3533, 2015.
[7] B. Cai, X. Xu, K. Jia, C. Qing, and D. Tao, "Dehazenet: An end-to-end system for single image haze removal," IEEE Transactions on Image Processing, vol. 25, no. 11, pp. 5187-5198, 2016.
[8] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," presented at the Proceedings of the IEEE conference on computer vision and pattern recognition.
[9] H. Zhang and V. M. Patel, "Densely connected pyramid dehazing network," presented at the Proceedings of the IEEE conference on computer vision and pattern recognition.
[10] X. Qin, Z. Wang, Y. Bai, X. Xie, and H. Jia, "FFA-Net: Feature Fusion Attention Network for Single Image Dehazing," arXiv preprint arXiv:1911.07559, 2019.
[11] S. Du, Y. Liu, M. Ye, Z. Xu, J. Li, and J. Liu, "Single image deraining via decorrelating the rain streaks and background scene in gradient domain," Pattern Recognition, vol. 79, pp. 303-317, 2018.
[12] X. Fu, J. Huang, D. Zeng, Y. Huang, X. Ding, and J. Paisley, "Removing rain from single images via a deep detail network," presented at the Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition.
[13] X. Jin, Z. Chen, and W. Li, "AI-GAN: Asynchronous interactive generative adversarial network for single image rain removal," Pattern Recognition, vol. 100, p. 107143, 2020.
[14] P. Liu, H. Zhang, K. Zhang, L. Lin, and W. Zuo, "Multi-level wavelet-CNN for image restoration," presented at the Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops.
[15] R. Qian, R. T. Tan, W. Yang, J. Su, and J. Liu, "Attentive generative adversarial network for raindrop removal from a single image," presented at the Proceedings of the IEEE conference on computer vision and pattern recognition.
[16] Y. Que, S. Li, and H. J. Lee, "Attentive Composite Residual Network for Robust Rain Removal from Single Images," IEEE Transactions on Multimedia.
[17] W. Yang, R. T. Tan, J. Feng, J. Liu, Z. Guo, and S. Yan, "Deep joint rain detection and removal from a single image," presented at the Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition.
[18] H. Zhang and V. M. Patel, "Density-aware single image de-raining using a multi-stream dense network," presented at the Proceedings of the IEEE conference on computer vision and pattern recognition.
[19] Y.-F. Liu, D.-W. Jow, S.-C. Hung, and J.-N. Hwang, "DesnowNet: Context-aware deep network for snow removal," IEEE Transactions on Image Processing, vol. 27, no. 6, pp. 3064-3073, 2018.
[20] Z. Li et al., "Single image snow removal via composition generative adversarial networks," IEEE Access, vol. 7, pp. 25016-25025, 2019.
[21] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, "Image-to-image translation with conditional adversarial networks," in Proceedings of the IEEE conference on computer vision and pattern recognition, 2017, pp. 1125-1134.
[22] D. Gupta and S. Choubey, "Discrete wavelet transform for image processing," International Journal of Emerging Technology and Advanced Engineering, vol. 4, no. 3, pp. 598-602, 2015.
[23] P. Liu, H. Zhang, W. Lian, and W. Zuo, "Multi-level wavelet convolutional neural networks," IEEE Access, vol. 7, pp. 74973-74985, 2019.
[24] T. Williams and R. Li, "Wavelet pooling for convolutional neural networks," 2018.
[25] S. Gao, M.-M. Cheng, K. Zhao, X.-Y. Zhang, M.-H. Yang, and P. H. Torr, "Res2net: A new multi-scale backbone architecture," IEEE transactions on pattern analysis and machine intelligence, 2019.
[26] G. Kaur and R. Kaur, "Image de-noising using wavelet transform and various filters," International Journal of Research in Computer Science, vol. 2, no. 2, p. 15, 2012.
[27] A. Jolicoeur-Martineau, "The relativistic discriminator: a key element missing from standard GANs," arXiv preprint arXiv:1807.00734, 2018.
[28] T. L. Sung and H. J. Lee, "Image-to-Image Translation Using Identical-Pair Adversarial Networks," Applied Sciences, vol. 9, no. 13, p. 2668, 2019.
[29] C. Wang, C. Xu, C. Wang, and D. Tao, "Perceptual adversarial networks for image-to-image transformation," IEEE Transactions on Image Processing, vol. 27, no. 8, pp. 4066-4079, 2018.
[30] H. Zhao, O. Gallo, I. Frosio, and J. Kautz, "Loss functions for image restoration with neural networks," IEEE Transactions on computational imaging, vol. 3, no. 1, pp. 47-57, 2016.
[31] S.-C. Pei, Y.-T. Tsai, and C.-Y. Lee, "Removing rain and snow in a single image using saturation and visibility features," presented at the
2014 IEEE International Conference on Multimedia and Expo Workshops (ICMEW),

[32] X. Zheng, Y. Liao, W. Guo, X. Fu, and X. Ding, "Single-image-based rain and snow removal using multi-guided filter," presented at the International Conference on Neural Information Processing,

[33] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, "Image-to-image translation with conditional adversarial networks," presented at the Proceedings of the IEEE conference on computer vision and pattern recognition,

[34] H. Huang, R. He, Z. Sun, and T. Tan, "Wavelet-smnet: A wavelet-based cnn for multi-scale face super resolution," presented at the Proceedings of the IEEE International Conference on Computer Vision,

[35] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," presented at the International Conference on Medical image computing and computer-assisted intervention,

[36] I. Goodfellow et al., "Generative adversarial nets," presented at the Advances in neural information processing systems,

[37] X. Mao, Q. Li, H. Xie, R. Y. Lau, Z. Wang, and S. Paul Smolley, "Least squares generative adversarial networks," presented at the Proceedings of the IEEE International Conference on Computer Vision,

[38] S. G. Mallat, "A theory for multiresolution signal decomposition: the wavelet representation," IEEE transactions on pattern analysis and machine intelligence, vol. 11, no. 7, pp. 674-693, 1989.

[39] R. Zhang, P. Isola, A. A. Efros, E. Shechtman, and O. Wang, "The unreasonable effectiveness of deep features as a perceptual metric," in Proceedings of the IEEE conference on computer vision and pattern recognition, 2018, pp. 586-595.

[40] Z. Wang, E. P. Simoncelli, and A. C. Bovik, "Multiscale structural similarity for image quality assessment," presented at the The Thirty-Seventh Asilomar Conference on Signals, Systems & Computers, 2003.

[41] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, "Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs," IEEE transactions on pattern analysis and machine intelligence, vol. 40, no. 4, pp. 834-848, 2017.