

Dense Scene Information Estimation Network for Dehazing

Tiantong Guo, Xuelu Li, Venkateswararao Cherukuri, Vishal Monga

The Pennsylvania State University, The Department of Electrical Engineering, University Park, PA, USA
 tiantong@ieee.org, xuelu@psu.edu, vmc5164@psu.edu, vmonga@engr.psu.edu

Abstract

Image dehazing continues to be one of the most challenging inverse problems. Deep learning methods have emerged to complement traditional model based methods and have helped define a new state of the art in achievable dehazed image quality. Yet, practical challenges remain in dehazing of real-world images where the scene is heavily covered with dense haze, even to the extent that no scene information can be observed visually. Many recent dehazing methods have addressed this challenge by designing deep networks that estimate physical parameters in the haze model, i.e. ambient light (\mathbf{A}) and transmission map (t). The inverse of the haze model may then be used to estimate the dehazed image. In this work, we develop two novel network architectures to further this line of investigation. Our first model, denoted as At-DH, designs a shared DenseNet based encoder and two distinct DenseNet based decoders to jointly estimate the scene information viz. \mathbf{A} and t respectively. This in contrast to most recent efforts (include those published in CVPR'18) that estimate these physical parameters separately. As a natural extension of At-DH, we develop the AtJ-DH network, which adds one more DenseNet based decoder to jointly recreate the haze-free image along with \mathbf{A} and t . The knowledge of (ground truth) training dehazed/clean images can be exploited by a custom regularization term that further enhances the estimates of model parameters \mathbf{A} and t in AtJ-DH. Experiments performed on challenging benchmark image datasets of NTIRE'19 and NTIRE'18 demonstrate that At-DH and AtJ-DH can outperform state-of-the-art alternatives, especially when recovering images corrupted by dense haze.

1. Introduction

Haze is an atmospheric phenomenon in which dust, smoke, and other particulates obscure the clarity. Haze can potentially incur dramatic degradation to the visibility of scenes captured, esp. in inclement weather. While hazing leads to a loss of quality for visual consumption, the presence of haze in digital images has further negative effects

on high-level computer vision tasks such as detection and recognition[1] of objects.

Dehazing refers to the technique of using algorithmic methods to process hazed images and produce (dehazed) images with high visual quality. The need for effective dehazing algorithms has been fueled by emerging applications in photography from mobile devices, autonomous driving and navigation systems, etc.

Many methods have been proposed to counteract the negative influence of haze and their starting point is the classical haze model introduced by [2] as below:

$$\mathbf{I} = \mathbf{J} \cdot \mathbf{t} + \mathbf{A} \cdot (1 - \mathbf{t}) \quad (1)$$

where \cdot represents element-wise multiplication, \mathbf{I} is the observed hazy image, \mathbf{J} is the true scene radiance, \mathbf{A} is the ambient light intensity, and \mathbf{t} is the transmission map. \mathbf{t} is a distance-dependent factor that affects the fraction of light which is able to reach the camera sensor. \mathbf{t} can be expressed as $\mathbf{t} = e^{-\beta d}$, where β represents the attenuation coefficient of the atmosphere and d represents the scene depth. The image dehazing task is essentially a process of recovering \mathbf{J} based on \mathbf{I} , which would inevitably lead to a heavily ill-posed problem according to Eq. (1). It can be observed from Eq. (1) that there are multiple possibilities for the solution choices for any given hazy image as the input. It is also clear that if the information about d is provided, the task of recovering \mathbf{J} will become much easier. This depth information is however rarely available in practice.

Image dehazing has been actively researched over the past two decades [3, 4, 5, 6, 7, 8, 9, 10, 11]. Existing work can be categorized into multi-image and single image dehazing. Restricted by the availability of the parameters that describe the scene information, much early research focused on multi-image dehazing [12, 13]. The availability of many images of the same scene under different weather/environmental conditions is however often unrealistic.

As a result, in recent efforts, single image dehazing has gained popularity [14, 15]. Most existing single image dehazing methods attempt to recover \mathbf{J} based on \mathbf{I} via the estimation of \mathbf{t} , including applying dark channel priors and color regularizers, etc. Performance benefits of

these techniques over early methods has been demonstrated [16, 17, 18, 19, 20, 21, 22].

Recently, Deep Learning (DL) methods have been shown to achieve success in imaging inverse problems such as image super-resolution [23, 24], deblurring [25], and inpainting [26]. For single image dehazing, DL methods usually require \mathbf{I} and \mathbf{J} as training pairs to learn mappings between \mathbf{I} and \mathbf{J} , \mathbf{I} and \mathbf{t} , and/or \mathbf{I} and \mathbf{A} . A detailed review of related literature is provided in Sec. 2. These methods take advantage of the powerful generalization ability of the deep networks to model the non-linear mapping function. However, as the real-world \mathbf{I} and \mathbf{J} pairs are hard to come by, synthetic datasets are often used during training.

In real-world scenes, the haze level can be much denser than is the case in most synthetic training datasets. The NTIRE2019-Dehaze Challenge dataset is representative of real-world challenges. Even if parts of the image are well recovered, no explicit visual information can be often be seen in areas with the densest haze.

In this paper, we propose a Dense Scene Information Estimation Network to directly tackle this issue. Our first model, denoted as At-DH, designs a shared DenseNet based encoder and two distinct DensenetNet based decoders to jointly estimate the scene information *viz.* \mathbf{A} and \mathbf{t} respectively. State of the art success in dehazing has been shown by approaches that use a custom designed deep network for estimating model parameters – such as [27] – however, they design *separate* networks to estimate \mathbf{A} and \mathbf{t} .

To incorporate more structural information in the learning process and overcome the difficulty of estimating \mathbf{t} and \mathbf{A} in dense haze, we develop an extension of At-DH called the AtJ-DH network, which adds one more DenseNet based decoder to jointly recreate the haze-free image along with \mathbf{A} and \mathbf{t} . The knowledge of (ground truth) training dehazed/clean images can be exploited by a custom regularization term that further enhances the estimates of model parameters \mathbf{A} and \mathbf{t} in AtJ-DH. Experiments performed on challenging benchmark image datasets of NTIRE’19 and NTIRE’18 demonstrate that At-DH and AtJ-DH can outperform state-of-the-art alternatives, especially when it comes to dense haze. Based on PSNR and SSIM results, the AtJ-DH *ranks 1st* and At-DH *ranks 3rd* in the NTIRE’19 Dehazing Challenge [28].

2. Related Work

Multiple deep learning based methods have been proposed to estimate \mathbf{t} and \mathbf{A} from \mathbf{I} to reconstruct \mathbf{J} . For example, Cai [29] introduced an end-to-end CNN network to estimate \mathbf{t} with a novel BReLU unit. More recently, Ren [30] also proposed a multi-scale deep neural network to estimate \mathbf{t} . However, these methods all have the limitation that they only put \mathbf{t} into consideration in their CNN frameworks without introducing any other essential information.

Li [1] proposed an all-in-one dehazing network to address the problem, where a linear transpose is leveraged to encode \mathbf{t} and \mathbf{A} into one variable.

The success of Generative Adversarial Networks (GANs) [31] in synthesizing realistic images has led researchers to explore more possibilities in the applications of GANs. In [32], to further incorporate the mutual structure information between the estimated \mathbf{t} and the dehazed result, the authors proposed a joint-discriminator based on GAN to decide whether the corresponding dehazed image and the estimated transmission map are real or fake. In [27], the authors present a multi-scale image dehazing method using perceptual pyramid deep network based on the recently popular dense blocks and residual blocks. The method involves an encoder-decoder structure with a pyramid pooling module in the decoder to incorporate contextual information of the scene while decoding. In [33], the authors proposed a bi-directional consistency loss to estimate \mathbf{t} and \mathbf{A} to reconstruct \mathbf{J} based on their specially designed fully convolutional neural network. In [34], a bilinear convolutional neural network is used to estimate \mathbf{t} and \mathbf{A} with the help of the proposed bilinear composition loss function which directly model the correlations between \mathbf{t} , \mathbf{A} and \mathbf{J} .

3. Scene Information Estimation Network

Our proposed deep network framework consists of a shared encoder and multiple decoders. The encoder serves as a general feature extractor and the decoders are trained to estimate the scene information based on the features extracted from the encoder. Inspired by the success of dense networks in various applications including dehazing [32], we build our encoder and decoders with DenseNet blocks.

3.1. Network Building Blocks

The proposed network is targeted to estimate the scene information from corrupted images with dense haze. The estimation system along with the refinement blocks are constructed by using the following building blocks:

- 1) Encoder: the encoder is constructed based on the Densely Connected Network (DCN) [35].
- 2) Decoder: the decoder has a similar structure as the encoder but with more batch normalization layers.
- 3) Refinement blocks: the refinement blocks in [32] is used for refine the outputs at different scales.

The encoder blocks from ‘Base.0’ to ‘Dense.4’ in Table 1 are initialized with pre-trained parameters from [35], which is originally trained for image classification. These connected pre-trained blocks also have the ability to obtain representative features in dehazing tasks. The dense blocks in the latter half of the DCN from [35] are not utilized since the features obtained through it will be mapped into a lower-dimension feature space only for classification. To make the pre-trained dense blocks be better applied for dehazing, we

Table 1: Scene Information Estimation Network Encoder Structure

	Base.0	Dense.1	Trans.1	Dense.2	Trans.2
Input Structure Output	input patch/image $\begin{bmatrix} 7 \times 7 \text{ conv.} \\ 3 \times 3 \text{ max-pool} \end{bmatrix}$ $64 \times 64 \times 64$	Base.0 $\begin{bmatrix} 1 \times 1 \text{ conv.} \\ 3 \times 3 \text{ conv.} \end{bmatrix} \times 6$ $64 \times 64 \times 256$	Dense.1 $\begin{bmatrix} 1 \times 1 \text{ conv.} \\ 2 \times 2 \text{ avg-pool} \end{bmatrix}$ $32 \times 32 \times 128$	Trans.1 $\begin{bmatrix} 1 \times 1 \text{ conv.} \\ 3 \times 3 \text{ conv.} \end{bmatrix} \times 12$ $32 \times 32 \times 512$	Dense.2 $\begin{bmatrix} 1 \times 1 \text{ conv.} \\ 2 \times 2 \text{ avg-pool} \end{bmatrix}$ $16 \times 16 \times 256$
Input Structure Output					
	Dense.3 $\begin{bmatrix} 1 \times 1 \text{ conv.} \\ 3 \times 3 \text{ conv.} \end{bmatrix} \times 24$ $16 \times 16 \times 1792$	Trans.3 $\begin{bmatrix} 1 \times 1 \text{ conv.} \\ 3 \times 3 \text{ conv.} \end{bmatrix} \times 6$ $8 \times 8 \times 896$	Dense.4 $\begin{bmatrix} 1 \times 1 \text{ conv.} \\ 2 \times 2 \text{ avg-pool} \end{bmatrix}$ $8 \times 8 \times 1344$	Trans.4 $\begin{bmatrix} 1 \times 1 \text{ conv.} \\ 3 \times 3 \text{ conv.} \end{bmatrix} \times 12$ $16 \times 16 \times 256$	

Table 2: Scene Information Estimation Decoder Structure

	Dense.5	Trans.5	Res.5	Dense.6	Trans.6	Res.6
Input Structure Output	[Res.4, Trans.2] $\begin{bmatrix} \text{batch norm} \\ 3 \times 3 \text{ conv.} \end{bmatrix} \times 7$ $16 \times 16 \times 832$	Dense.5 $\begin{bmatrix} 1 \times 1 \text{ conv.} \\ \text{upsample 2} \end{bmatrix}$ $32 \times 32 \times 160$	Trans.5 $\begin{bmatrix} 3 \times 3 \text{ conv.} \\ 3 \times 3 \text{ conv.} \end{bmatrix} \times 2$ $32 \times 32 \times 160$	[Trans.1, Res.5] $\begin{bmatrix} \text{batch norm} \\ 3 \times 3 \text{ conv.} \end{bmatrix} \times 7$ $32 \times 32 \times 416$	Dense.6 $\begin{bmatrix} 1 \times 1 \text{ conv.} \\ \text{upsample 2} \end{bmatrix}$ $64 \times 64 \times 64$	Trans.6 $\begin{bmatrix} 3 \times 3 \text{ conv.} \\ 3 \times 3 \text{ conv.} \end{bmatrix} \times 2$ $64 \times 64 \times 64$
Input Structure Output	Dense.7 $\begin{bmatrix} \text{batch norm} \\ 3 \times 3 \text{ conv.} \end{bmatrix} \times 7$ $64 \times 64 \times 128$	Trans.7 $\begin{bmatrix} 1 \times 1 \text{ conv.} \\ \text{upsample 2} \end{bmatrix}$ $128 \times 128 \times 32$	Res.7 $\begin{bmatrix} 3 \times 3 \text{ conv.} \\ 3 \times 3 \text{ conv.} \end{bmatrix} \times 2$ $128 \times 128 \times 32$	Dense.8 $\begin{bmatrix} \text{batch norm} \\ 3 \times 3 \text{ conv.} \end{bmatrix} \times 7$ $128 \times 128 \times 64$	Trans.8 $\begin{bmatrix} 1 \times 1 \text{ conv.} \\ \text{upsample 2} \end{bmatrix}$ $256 \times 256 \times 16$	Res.8 $\begin{bmatrix} 3 \times 3 \text{ conv.} \\ 3 \times 3 \text{ conv.} \end{bmatrix} \times 2$ $256 \times 256 \times 16$
Input Structure Output	Refine.9 [Input, Res.8]	Refine.10 $\begin{bmatrix} 32 \times 32 \text{ avg-pool} \\ 1 \times 1 \text{ conv.} \\ \text{upsample} \end{bmatrix}$ $256 \times 256 \times 1$	Refine.11 $\begin{bmatrix} 16 \times 16 \text{ avg-pool} \\ 1 \times 1 \text{ conv.} \\ \text{upsample} \end{bmatrix}$ $256 \times 256 \times 1$	Refine.12 $\begin{bmatrix} 8 \times 8 \text{ avg-pool} \\ 1 \times 1 \text{ conv.} \\ \text{upsample} \end{bmatrix}$ $256 \times 256 \times 1$	Refine.13 $\begin{bmatrix} 4 \times 4 \text{ avg-pool} \\ 1 \times 1 \text{ conv.} \\ \text{upsample} \end{bmatrix}$ $256 \times 256 \times 1$	Output.14 [Refine.9.10.11.12.13] $256 \times 256 \times X^1$

append the encoder with newly added blocks ‘Trans.4’ and ‘Res.4’ to enlarge the features generated by ‘Dense.4’. It can help the encoder preserve more spatial feature information from the pre-trained dense blocks which is then utilized by decoders for further enhancement in dehazing.

Table 2 provides more details about the structure of the decoder. The decoder is built to exploit the extracted features from the encoder for dehazing. It contains 4 ‘Trans’ blocks, which stand for transformation block. Each transformation block essentially plays the role of reordering and enlarging the refined image/features. The reordering process is accomplished by a 1×1 convolutional layer, and the enlargement is done by the upsampling layer². We added new residual blocks [36] in between two successive dense blocks to incorporate more high-frequency information which can assist in recovering more details in the recovered image. In addition to that, batch normalization layers are added to the dense blocks to normalize the training data so that the manifold of the network parameters will be smoothed and the network will enjoy better training stability[37].

The bottom row of Table 2 details the structure of refinement blocks as suggested by [27, 32]. These blocks first use

¹The number of the final output channel is decided by the functionality of each decoder. See Table 3 and 4 for details.

²the upsampling layer can be replaced by a transpose conv. layer.

average pooling layers at spatial size of 32×32 , 16×16 , 8×8 , and 4×4 to extract local average information. Then, the 1×1 convolutional layer refines the outputs and an up-sampling layer is used to enlarge the image into the desired spatial size. These enlarged locally reorganized images are then appended together and passed through the final refinement layer which uses several convolutional layers to eliminate the blocking artifacts. This refinement practice would allow the image information to be merged and retouched at different scales.

3.2. At-DH Network

With the aforementioned building blocks, the proposed At-DH network structure is described in Table 3 and shown in the dash-lined part in Fig. 1. The encoder in At-DH is constructed as described in Table 1 and the Decoder.A and Decoder.t are constructed as illustrated in Table 2. The features extracted by the encoder are densely connected to the decoder. As shown in Table 2, ‘Dense.5’ utilizes the concatenated outputs from ‘Res.4’ and ‘Trans.2’ to have better information flow as suggested in [35]. Similar connection is used in ‘Dense.6’ as well.

In the At-DH model, the two decoders are trained to generate the $\hat{\mathbf{A}}$ and $\hat{\mathbf{t}}$ such that combined with the input hazy image \mathbf{I} , the At-DH network can generate haze-free images

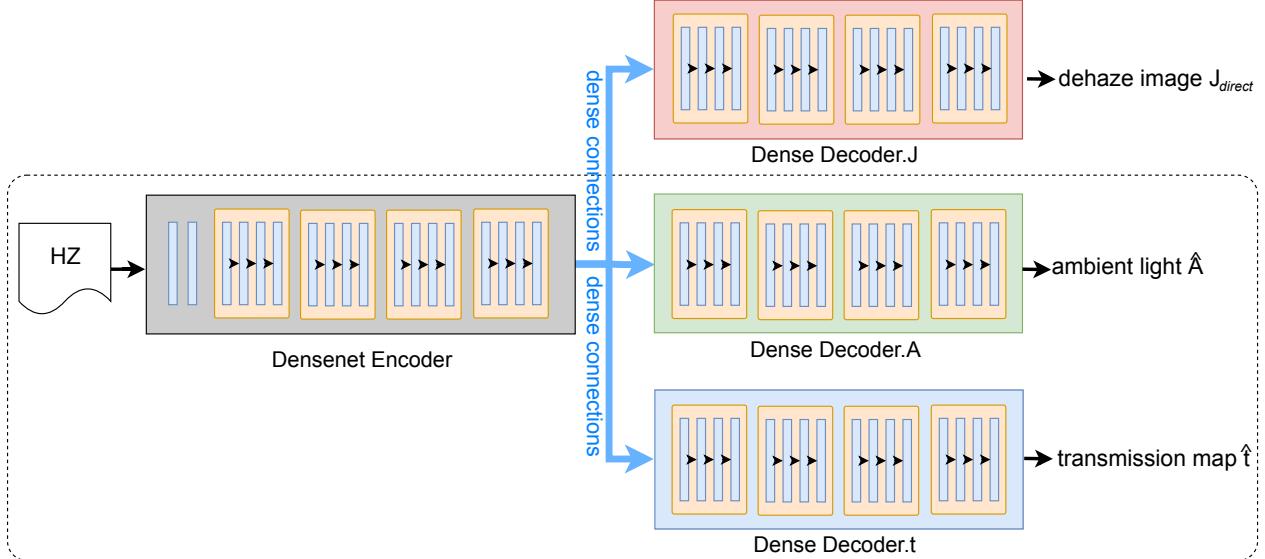


Figure 1: The proposed the At-DH (within dashlines) and AtJ-DH network. A shared encoder is employed to extract image features that are subsequently exploited by the two decoders (Decoder.A and Decoder.t) to jointly estimate the physical model parameters. AtJ-DH employs an additional decoder, i.e. Decoder.J which estimates the haze-free image. Employing custom regularization in the learning of AtJ-DH using the ground truth \mathbf{J} further enhances the estimates $\hat{\mathbf{A}}$ and $\hat{\mathbf{t}}$. The architectural details of the encoder and decoder are listed in Table 1, 2. At-DH and AtJ-DH network architectures are listed Tabel 3 and 4.

Table 3: At-DH Model Structure

	Encoder	Decoder.A	Decoder.t
Input Structure	Input	Encoder	Encoder
	As in Table 1	As in Table 2 with $X = 3$	As in Table 2 with $X = 1$

Table 4: AtJ-DH Model Structure

	Encoder	Decoder.A	Decoder.t	Decoder.J
Input Structure	Input	Encoder	Encoder	Encoder
	As in Table 1	As in Table 2 with $X = 3$	As in Table 2 with $X = 1$	As in Table 2 with $X = 3$

by the inverse haze model as following:

$$\hat{\mathbf{J}} = \frac{\mathbf{I} - \hat{\mathbf{A}} \cdot (1 - \hat{\mathbf{t}})}{\hat{\mathbf{t}}} \quad (2)$$

where $\hat{\mathbf{J}}$ is the reconstructed haze-free image, $\hat{\mathbf{A}} := f_A(\mathbf{I})$ and $\hat{\mathbf{t}} := f_t(\mathbf{I})$. $f_A(\cdot)$ and $f_t(\cdot)$ denote the outputs of Decoder.A and Decoder.t, respectively. We can also reconstruct the hazy image using the decoders' outputs and the haze-free image during the *training* by the inverse haze-model as following:

$$\hat{\mathbf{I}} = \mathbf{J} \cdot \hat{\mathbf{t}} - \hat{\mathbf{A}} \cdot (1 - \hat{\mathbf{t}}) \quad (3)$$

where $\hat{\mathbf{I}}$ is the self-reconstructed hazy image. The training of these two decoders is accomplished by minimizing the following loss function:

$$\mathcal{L} = \mathcal{L}_{\ell_2} + \alpha \mathcal{L}_{\text{vgg}} \quad (4)$$

where the \mathcal{L}_{ℓ_2} is the (re)construction loss between the hazy images (\mathbf{I} and $\hat{\mathbf{I}}$) and the haze-free images (\mathbf{J} and $\hat{\mathbf{J}}$) during the training:

$$\mathcal{L}_{\ell_2} = \|\hat{\mathbf{J}} - \mathbf{J}\|_2^2 + \|\hat{\mathbf{I}} - \mathbf{I}\|_2^2 \quad (5)$$

where the $\hat{\mathbf{J}}$ and $\hat{\mathbf{I}}$ are constructed following Eqs. (2) and (3). Another important loss term in Eq. (4) is the perceptual loss, which is used to ensure that the overall image content is agree with the ground-truth image. To achieve this, the

high-level features extracted from the pre-trained VGG network are used. The VGG based regularizer is given by:

$$\mathcal{L}_{\text{vgg}} = \sum_{i=1}^3 \|g_i(\hat{\mathbf{J}}) - g_i(\mathbf{J})\|_2^2 + \sum_{i=1}^3 \|g_i(\hat{\mathbf{I}}) - g_i(\mathbf{I})\|_2^2 \quad (6)$$

where the $g_i(\cdot)$ represents the operator formed by the pre-trained feature extraction layers in the vgg16 model that ends with layers ReLu1_1, ReLu2_2, and ReLu3_3. The \mathcal{L}_{ℓ_2} and \mathcal{L}_{vgg} are balanced by the parameter α .

3.3. AtJ-DH Network

The At-DH with self-regularization generates dehazed images with enhanced visual quality. However, for areas in images where the haze is so dense that any scene information in \mathbf{I} is visually obscured, the two decoders for \mathbf{A} and \mathbf{t} do not suffice. Hence, we extend the proposed 'At' model by introducing a new J-decoder. Since the J-decoder is trained by treating the hazy observation and corresponding ground truth as training pairs, it is able to provide real scene information during the training process.

Our proposed AtJ-DH network is visually illustrated in Fig. 1 and its architecture is detailed in Table 4. Decoder.A and Decoder.t are trained to estimate the \mathbf{A} and \mathbf{t} in the haze model while Decoder.J is trained to directly recover

the haze-free image. The outputs of Decoder.J are concatenated together and sent to the refinement block which is described as Refine.10 to Refine.13 units in Table 2. Several convolutional layers are used to eliminate blocking artifacts that may be generated by the refinement blocks.

In the AtJ-DH network, the outputs of the three decoders are defined as $\hat{\mathbf{A}} := f_A(\mathbf{I})$, $\hat{\mathbf{t}} := f_t(\mathbf{I})$ and

$$\mathbf{J}_{direct} := f_J(\mathbf{I}) \quad (7)$$

Compared with the At-DH network, the AtJ-DH network has an additional Decoder.J (Eq. (7)), whose job is to recover the haze-free image directly. By adding Decoder.J, the network would have the ability to make reconstructed scene information be more consistent with ground truth. The separate network system formed by the encoder and Decoder.J make it possible for Decoder.J to generate real scene details even in areas which are full of dense haze.

During the training, Decoder.J provides further guidelines for Decoder.A and Decoder.t to produce better estimations of the scene information. The training of the AtJ-DH is regularized by three loss terms:

- Training Decoder.J by minimizing:

$$\mathcal{L}_J = \|\mathbf{J}_{direct} - \mathbf{J}\|_2^2 + \beta \sum_{i=1}^3 \|g_i(\mathbf{J}_{direct}) - g_i(\mathbf{J})\|_2^2 \quad (8)$$

where the first term is the standard ℓ_2 loss function that is commonly used in regression problems. The second term is the VGG based perceptual loss similar to Eq. (6). With the help of this loss function, Decoder.J essentially learns to recreate the haze-free image based on the hazy input. The ground truth information introduced by training Decoder.J will be later utilized to guide Decoder.A and Decoder.t to better estimate \mathbf{A} and \mathbf{t} which preserve the scene information. β is a weight parameters that controls the relative importance of the loss terms.

- Training Decoder.A and Decoder.t by minimizing:

$$\begin{aligned} \mathcal{L}_{At} = & \|\hat{\mathbf{J}} - \mathbf{J}\|_2^2 + \gamma \|\hat{\mathbf{I}} - \mathbf{I}\|_2^2 \\ & + \kappa \sum_{i=1}^3 \|g_i(\hat{\mathbf{J}}) - g_i(\mathbf{J})\|_2^2 + \kappa \sum_{i=1}^3 \|g_i(\hat{\mathbf{I}}) - g_i(\mathbf{I})\|_2^2 \end{aligned} \quad (9)$$

where $\hat{\mathbf{I}}$ and $\hat{\mathbf{J}}$ are given in Eq. (3) and (2). The \mathcal{L}_{At} is the same as the loss function of At-DH network, which consists of the ℓ_2 reconstruction loss and the vgg perceptual loss. The γ and κ serve as the trade-off parameters to balance different loss terms selected by cross-validation [38].

- Training Decoder.A, Decoder.t, and Decoder.J together by minimizing:

$$\mathcal{L}_{AtJ} = \|\mathbf{J}_{direct} \cdot \hat{\mathbf{t}} + (1 - \hat{\mathbf{t}}) \cdot \hat{\mathbf{A}} - \mathbf{I}\|_2^2 \quad (10)$$

As we discussed above, when the haze is too dense, it will be difficult to use Decoder.A and Decoder.t alone to estimate all the scene information. By combining with Decoder.J, Decoder.A and Decoder.t can be learned to re-



(a) Haze free image (b) NTIRE18 hazy image (c) Synthetic dense haze
Figure 2: NTIRE18 image 27-indoor.png(upper) and 36-outdoor.png, the synthetic dense-haze image adds more haze based on the NTIRE18 hazy image.

construct the hazy image based on the guidance from the ground truth information introduced by training Decoder.J. It is noted that Decoder.J, A, and t are trained jointly at each step so that the learned information from all three decoders can be integrated for next training steps.

For both networks, the final dehazed images are generated using the estimated \mathbf{t} and \mathbf{A} following Eq. (2).

4. Dataset, Training, and Test Procedure

4.1. Datasets

To train At-DH and AtJ-DH, we use the NTIRE2019-Dehaze dataset [39]. The images were collected by a professional camera including professional fog generators, so as to capture the same scene under both conditions (with and without haze). The training data consists of 45 hazy images (with dense haze generated for both indoor and outdoor environments) and their corresponding ground truth (haze-free) images with exactly the same scene information.

To learn a network with more powerful generalization ability, we also include the NTIRE2018-Dehaze dataset [40] in training. Compared to the NTIRE2018 dataset, the haze is much denser in the NTIRE2019 dataset. We developed a synthetic method to thicken the haze in NTIRE2018 training images in order to reach the same haze level as in the NTIRE2019. We generated the synthetic dense-haze image following the practice demonstrated in [29] as below:

$$\mathbf{I}_{syn} = \mathbf{I} \cdot \mathbf{t} + (1 - \mathbf{t}) \cdot \mathbf{A} \quad (11)$$

where, \mathbf{I}_{syn} is the synthetic dense-haze image, \mathbf{I} is the NTIRE2018 hazy image. \mathbf{A} is chosen to be a fixed value to reduce the uncertainty in variable learning. For indoor images, we set $\mathbf{A} = [0.6, 0.6, 0.6]$, and for outdoor images we set $\mathbf{A} = [0.80, 0.81, 0.86]^3$ to give a blueish atmosphere light which is estimated by using method described in [21]. The \mathbf{t} is selected uniformly between 0.01 and 0.3 to add dense haze as smaller \mathbf{t} yields more atmosphere information. These values of \mathbf{A} and \mathbf{t} are chosen according to the results of the cross-validation during the development phase. As shown in Fig. 2, the aforementioned synthetic method can generate images with dense haze. During the training, patches of size 512×512 are extracted from the training images. The augmentations are used as the combination of the following options: 1) horizontal flip, rotation

³For RGB channels, \mathbf{A} is set to 0.80, 0.81, and 0.86 respectively.

by 90° , 180° , and 270° ; 2) scale to 0.7, 0.8, and 0.9 of the original image size. Also, the images (whole image, not patches) are resized to 512×512 and applied the same augmentation strategies on these resized images and included them for training.

4.2. Training

Learning the complete network for once is challenging and can result instability in training as the network is highly parameterized. Therefore, we adapted a two-stage training strategy as described below:

Stage 1 - Pre-training of Encoder: We first pre-train the encoder by combining it with a single decoder to make that the output of the decoder can be reconstructed as the ground truth. The structure of the encoder and the decoder is the same as described in Tables 1 and 2. In this stage, we used the NTIRE19 and synthetically dense-haze images from NTIRE18 to train the encoder for 80 epochs.

Stage 2 - At/AtJ-DH training: In this stage we combine the encoder trained from stage 1 with newly constructed decoders for estimating \mathbf{A} , \mathbf{t} , and \mathbf{J} . For At-DH, the decoders are as shown in Table 3 and trained using the losses detailed in Section 3.2. For AtJ-DH, the decoders are as shown in Table 4 and trained using the losses detailed in Section 3.3. The training is conducted on two datasets. For the first 50 epochs the training data is the same as in Stage 1. For the next 70 epochs, the training data is only from NTIRE19.

Adam optimizer [41] with initial learning rate of 1×10^{-4} is used for training. The learning rate is reduced to its 70% for every 35 epochs and reset in stage 2.

4.3. IRCNN Post-processing

While At/AtJ-DH is already able to recover scene information from the dense-haze input, we also used an *optional* IRCNN [42] denoiser with $\sigma = 15$ to further improve the results visually. IRCNN method combines the benefits of both model based and learning based techniques for image restoration applications. In this dehazing problem, we use the pre-trained CNN denoiser and incorporate it as a post processing unit after the output obtained by our proposed At/AtJ-DH framework. We use At/AtJ-DH+ to indicate that the post processing is present. Detailed results obtained via At/AtJ-DH and At/AtJ-DH+ are listed in Section 5.2.

5. Experimental Results

In this section we present the experimental results of our proposed At/AtJ-DH, that is, ablation study and comparison w.r.t state-of-the-art methods. The evaluation metrics used to quantify the performance are Peak Signal-to-Noise Ratio (PSNR) and Structural Similarity Index (SSIM) [43]⁴.

⁴Code is available at the project page: <http://signal.ee.psu.edu/research/ATJDH.html>

5.1. Ablation Study

The performances of different configurations of the proposed scene estimation network are investigated in this section. Table 6, 5, and 7 report the results of At-DH and AtJ-DH methods. As shown in the tables, by adding Decoder.J, AtJ-DH further improves both the PSNR and SSIM scores, and produces visually pleasing images (see Fig. 5).

5.2. Comparison with State-of-the-art Methods

This section illustrates the comparisons between our proposed methods with the state-of-the-art methods on real-world benchmark data sets I-HAZE, and O-HAZE [44, 45].

State-of-the-art Methods The state-of-the-art methods included in the comparisons are: CVPR'09 [17, 46], TIP'15 [47], ECCV'16 [30], TIP'16 [29], CVPR'16 [22], ICCV'17 [1], CVPR'18 [27], and CVPRW'18 [32].

Evaluation Datasets The comparisons are conducted on the I-HAZE (indoor) and O-HAZE (outdoor) validation datasets [40]. Each of the dataset contains 5 pairs of haze and haze-free image pairs. Detailed acquisition methods of these real-world hazy image pairs are discussed in [40].

Fig. 3 and 4 show the experimental results of the state-of-the-art methods compared with AtJ-DH conducted on NTIRE2018 indoor and outdoor validation datasets. It can be found that AtJ-DH generated much more visually pleasing results. As shown in Table 5 and 6, At/AtJ-DH outperforms other state-of-the-art methods when evaluated on PSNR and SSIM. With the help of the post processing procedure described in Section 4.3, the final dehazed images generated by AtJ-DH+ can obtain the highest scores.

5.3. NTIRE-2019 Dehazing Challenge

The haze presented in images from the NTIRE2019-Dehaze dataset is much denser than images in the previous literature. As shown in Fig. 5, the state-of-the-art methods' performances drop largely when applied to the dataset due to the reason that the dense haze has covered almost all the scene information. Since AtJ-DH can estimate more accurate scene information from the dense-haze image, the dehazed images generated by AtJ-DH are much more visually pleasing. We evaluate the quantitative performances of the methods on the NTIRE2019 validation set since their ground-truth images are made available [39]. As shown in Table 7, AtJ-DH outperforms all the other state-of-the-art methods. If the post processing procedure described in Section 4.3 is adapted, the final dehazed images generated by AtJ-DH+ can achieve the highest PSNR and SSIM scores.

Table 8 includes the top-6 methods from the contest. It is found that At-DH/+ and AtJ-DH/+ are among the top performing methods in the NTIRE2019-Dehazing Challenge and outperform other methods by a noticeable margin.

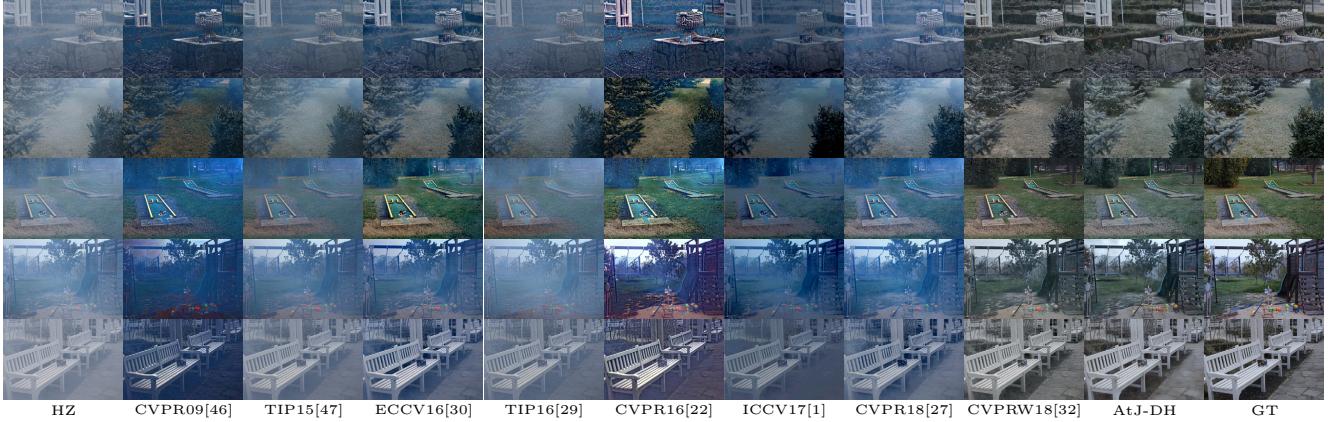


Figure 3: The visual results of NTIRE2018-outdoor validation dataset.



Figure 4: The visual results of NTIRE2018-indoor validation dataset.

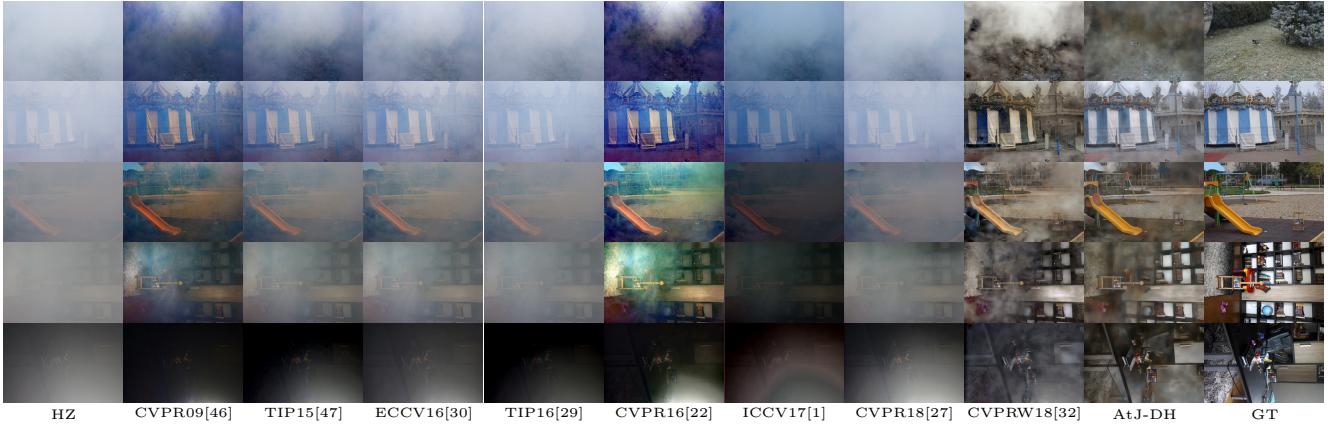


Figure 5: The visual results of NTIRE2019 validation dataset.

6. Conclusion

We focus on developing deep learning architectures that estimate physical parameters in the haze model jointly as opposed to separately. Our At-DH network, uses a shared DenseNet encoder and two distinct DenseNet decoders to jointly estimate the scene information *viz.* \mathbf{A} and \mathbf{t} . As a natural extension of At-DH, we develop the AtJ-DH network, which adds one more DenseNet decoder to jointly recreate the haze-free image along with \mathbf{A} and \mathbf{t} . Experi-

ments performed on challenging benchmark image datasets of NTIRE'19 and NTIRE'18 demonstrate that At(J)-DH can outperform state-of-the-art alternatives. Notably, in NTIRE'19 results At(J)-DH methods offer nearly a 1 dB gain in PSNR over the next best method. In AtJ-Dh we recover the final dehazed image using enhanced estimates of \mathbf{A} and \mathbf{t} in an inverse haze model; this dehazed image may be fused intelligently with the output of Decoder.J in the AtJ-DH network to further enhance performance.

Table 5: The PSNR/SSIM of different methods over NTIRE2018-outdoor validation dataset.

method	36.png	37.png	38.png	39.png	40.png	avg.
CVPR09 [46]	18.1820/0.4474	16.0912/0.4983	14.1227/0.0835	12.8787/0.3575	14.2106/0.3864	15.0970/0.3546
TIP15 [47]	17.4660/0.4976	16.1686/0.4533	15.1391/0.1796	14.7964/0.4131	16.3732/0.5683	15.9887/0.4224
TIP16 [30]	16.5891/0.4862	15.7593/0.4334	13.2500/0.1890	12.7816/0.3935	16.5339/0.5597	14.9828/0.4123
CVPR16 [29]	16.9236/0.4267	14.9854/0.4776	15.5448/0.3390	17.6496/0.4751	17.0424/0.5350	16.4292/0.4507
ICCV17 [1]	17.0951/0.4516	16.4676/0.3886	16.1153/0.1194	15.0439/0.3388	15.9477/0.5043	16.1339/0.3606
CVPR18 [27]	17.1374/0.4385	15.2847/0.4173	14.6555/0.1143	15.2353/0.3530	17.7805/0.5198	16.0187/0.3686
CVPRW18 [32]	24.6703/0.7288	22.4079/0.6551	23.7469/0.7199	21.9055/0.6296	22.2878/0.6822	23.0037/0.6831
At-DH	25.0979/0.8050	23.4639/0.7627	24.8048/0.8053	22.0375/0.7519	24.8303/0.8044	24.0469/0.7859
At-DH+	25.1437/0.8213	23.4302/0.7598	24.9171/0.8187	22.0731/0.7679	24.8260/0.8111	24.0780/0.7958
AtJ-DH	27.0772/0.8154	24.0295/0.7513	23.9662/0.7991	22.5974/0.7555	24.4090/0.8025	24.4159/0.7847
AtJ-DH+	27.1721/0.8316	23.9785/0.7455	24.0495/0.8076	22.6578/0.7707	24.4443/0.8107	24.4604/0.7932

Table 6: The PSNR/SSIM of different methods over NTIRE2018-indoor validation dataset.

method	26.png	27.png	28.png	29.png	30.png	avg.
CVPR09 [46]	8.2706/0.3545	12.8863/0.2014	12.7162/0.5485	12.1518/0.5411	13.4688/0.2753	11.8988/0.3842
TIP15 [47]	13.1816/0.6581	16.6858/0.3952	11.5135/0.5590	17.1496/0.7803	15.7567/0.3215	14.8574/0.5428
TIP16 [30]	10.1699/0.5498	14.5147/0.3094	13.3890/0.6349	11.9041/0.5369	15.5312/0.3412	13.1018/0.4744
CVPR16 [29]	12.4147/0.4800	14.7990/0.3639	13.2925/0.5489	14.6639/0.5296	13.9293/0.4057	13.8199/0.4656
ICCV17 [1]	10.8313/0.6185	16.8387/0.3943	12.7391/0.4692	15.3688/0.8054	17.2741/0.3095	14.6104/0.5194
CVPR18 [27]	15.3106/0.6283	16.0856/0.3512	9.8470/0.5540	22.2085/0.8013	15.4517/0.1977	15.7807/0.5065
CVPRW18 [32]	14.2680/0.6778	20.8952/0.7533	18.4479/0.6983	20.5845/0.8154	16.4299/0.5445	18.1251/0.6978
At-DH	20.5938/0.8760	22.9991/0.8490	19.9912/0.8313	22.9211/0.9001	18.5186/0.8056	21.0048/0.8524
At-DH+	20.6302/0.9015	23.0601/0.8696	20.0012/0.8490	22.9313/0.9200	18.5464/0.8335	21.0338/0.8747
AtJ-DH	21.8144/0.8917	23.3963/0.8630	20.8504/0.8439	24.8057/0.9218	20.5668/0.8319	22.2867/0.8705
AtJ-DH+	21.8553/0.9125	23.4660/0.8798	20.8737/0.8622	24.8174/0.9366	20.6111/0.8556	22.3247/0.8894

Table 7: The PSNR/SSIM of different methods over NTIRE2019 validation dataset.

method	46.png	47.png	48.png	49.png	50.png	avg.
CVPR09 [46]	10.2792/-0.0030	14.7181/0.3669	14.1058/0.2821	10.4352/0.2112	11.0434/0.3764	12.1163/0.2467
TIP15 [47]	9.3752/0.0187	16.1040/0.4152	13.4858/0.2282	10.9173/0.2735	11.2999/0.2600	12.2364/0.2391
TIP16 [30]	8.1189/0.0432	12.8189/0.3496	12.3575/0.2172	10.5432/0.2403	10.5051/0.2119	10.8687/0.2124
CVPR16 [29]	9.5131/0.0130	13.9106/0.3744	12.9015/0.2796	10.1643/0.2208	11.6160/0.3645	11.6211/0.2505
ICCV17 [1]	10.1190/-0.0160	14.1507/0.2835	12.1667/0.1804	8.4235/0.2104	12.8605/0.3342	11.5441/0.1985
CVPR18 [27]	8.0126/0.0288	13.0654/0.3498	13.5508/0.1945	11.0016/0.2758	14.0006/0.4475	11.9262/0.2593
CVPRW18 [32]	8.8668/0.2272	15.1229/0.3635	15.6723/0.4603	11.8703/0.3592	15.0124/0.4305	13.3090/0.3681
At-DH	16.0246/0.2641	20.6051/0.7221	19.3801/0.5365	12.7537/0.4482	15.8842/0.5935	16.9296/0.5129
At-DH+	16.0565/0.2809	20.6645/0.7435	19.4310/0.5657	12.7574/0.4677	15.8772/0.6122	16.9573/0.5340
AtJ-DH	16.4984/0.2690	20.8886/0.7242	19.7148/0.5382	12.9908/0.4562	16.4970/0.6086	17.1479/0.5226
AtJ-DH+	16.5354/0.2848	20.9509/0.7449	19.7752/0.5667	12.9963/0.4749	16.4918/0.6263	17.1899/0.5395

Table 8: The average PSNR/SSIM of top methods over NTIRE2019 test dataset.

team	contest method	PSNR	SSIM
ours	AtJ-DH+	20.258	0.657
	AtJ-DH	20.212	0.635
	At-DH+	19.528	0.642
	At-DH	19.479	0.615
other teams	method1	19.469	0.652
	method2	18.521	0.640

References

- [1] B. Li et al., “Aod-net: All-in-one dehazing network,” in *Proc. IEEE Conf. on Comp. Vis.*, 2017.
- [2] E. J. McCartney, “Optics of the atmosphere: scattering by molecules and particles,” *New York, John Wiley and Sons, Inc.*, 1976. 421 p., 1976.
- [3] C. O. Ancuti, C. Ancuti, and C. D. Vleeschouwer, “Effective local airlight estimation for image dehazing,” in *Proc. IEEE Conf. on Image Proc.*, 2018, pp. 2850–2854.
- [4] C. O. Ancuti et al., “Color transfer for underwater dehazing and depth estimation,” in *Proc. IEEE Conf. on Image Proc.*, 2017, pp. 695–699.
- [5] C. O. Ancuti et al., “Locally adaptive color correction for underwater image dehazing and matching,” in *Proc. IEEE Conf. on Image Proc.*, 2017, pp. 1–9.
- [6] C. O. Ancuti and C. Ancuti, “Single image dehazing by multi-scale fusion authors,” *IEEE Trans. on Image Proc.*, vol. 22, no. 8, pp. 3271–3282, 2013.
- [7] C. O. Ancuti, C. Ancuti, and P. Bekaert, “Effective single image dehazing by fusion,” in *Proc. IEEE Conf. on Image Proc.*, 2010, pp. 3541–3544.
- [8] C. Ancuti et al., “Night-time dehazing by fusion,” in *Proc. IEEE Conf. on Image Proc.*, 2016, pp. 2256–2260.
- [9] W. Ren et al., “Gated fusion network for single image dehazing,” in *Proc. IEEE Conf. on Comp. Vis. Patt. Recog.*, 2018, pp. 3253–3261.
- [10] C. Chen, M. N. Do, and J. Wang, “Robust image and video dehazing with visual artifact suppression via gradient resid-

- ual minimization,” in *Proc. IEEE European Conf. on Comp. Vision*. Springer, 2016, pp. 576–591.
- [11] R. Li et al., “Single image dehazing via conditional generative adversarial network,” in *Proc. IEEE Conf. on Comp. Vis. Patt. Recog.*, 2018, pp. 8202–8211.
- [12] S. G. Narasimhan and S. K. Nayar, “Chromatic framework for vision in bad weather,” in *Proc. IEEE Conf. on Comp. Vis. Patt. Recog.*, 2000, vol. 1, pp. 598–605.
- [13] Z. Li et al., “Simultaneous video defogging and stereo reconstruction,” in *Proc. IEEE Conf. on Comp. Vis. Patt. Recog.*, 2015, pp. 4988–4997.
- [14] S. G. Narasimhan and S. K. Nayar, “Interactive (de) weathering of an image using physical models,” in *Proc. IEEE Workshop Color and Photometric Methods in Comp. Vision*. France, 2003, vol. 6, p. 1.
- [15] N. Hautière et al., “Towards fog-free in-vehicle vision systems through contrast restoration,” in *Proc. IEEE Conf. on Comp. Vis. Patt. Recog.*, 2007, pp. 1–8.
- [16] R. T. Tan, “Visibility in bad weather from a single image,” in *Proc. IEEE Conf. on Comp. Vis. Patt. Recog.*, 2008.
- [17] K. He, J. Sun, and X. Tang, “Single image haze removal using dark channel prior,” *IEEE Trans. on Pattern Analysis and Machine Int.*, vol. 33, no. 12, pp. 2341–2353, 2011.
- [18] L. Kratz and K. Nishino, “Factorizing scene albedo and depth from a single foggy image,” in *Proc. IEEE Conf. on Comp. Vision*, 2009, pp. 1701–1708.
- [19] C. O. Ancuti, C. Ancuti, and P. Bekaert, “Effective single image dehazing by fusion,” in *Proc. IEEE Conf. on Image Proc.*, 2010, pp. 3541–3544.
- [20] G. Meng et al., “Efficient image dehazing with boundary constraint and contextual regularization,” in *Proc. IEEE Conf. on Comp. Vision*, 2013, pp. 617–624.
- [21] R. Fattal, “Dehazing using color-lines,” *ACM Trans. on Graphics*, vol. 34, no. 1, pp. 13, 2014.
- [22] D. Berman, T. Treibitz, and S. Avidan, “Non-local image dehazing,” in *Proc. IEEE Conf. on Comp. Vis. Patt. Recog.*, 2016, pp. 1674–1682.
- [23] T. Guo et al., “Deep wavelet prediction for image super-resolution,” in *Proc. IEEE Conf. Workshop on Comp. Vis. Patt. Recog.*, 2017.
- [24] T. Guo et al., “Orthogonally regularized deep networks for image super-resolution,” in *Proc. IEEE Int. on Conf. Acoustics, Speech, and Signal Proc.*, 2018, pp. 1463–1467.
- [25] S. Nah et al., “Deep multi-scale convolutional neural network for dynamic scene deblurring,” in *Proc. IEEE Conf. on Comp. Vis. Patt. Recog.*, 2017, pp. 3883–3891.
- [26] J. Xie, L. Xu, and E. Chen, “Image denoising and inpainting with deep neural networks,” in *Proc. Advances in Neural Information Proc. Systems*, 2012, pp. 341–349.
- [27] H. Zhang and V. M. Patel, “Densely connected pyramid dehazing network,” in *Proc. IEEE Conf. on Comp. Vis. Patt. Recog.*, 2018, pp. 3194–3203.
- [28] C. O. Ancuti, C. Ancuti, and R. T. et al., “Ntire 2019 challenge on image dehazing: Methods and results,” in *Proc. IEEE Conf. Workshop on Comp. Vis. Patt. Recog.*, 2019.
- [29] B. Cai et al., “Dehazenet: An end-to-end system for single image haze removal,” *IEEE Trans. on Image Proc.*, vol. 25, no. 11, pp. 5187–5198, 2016.
- [30] W. Ren et al., “Single image dehazing via multi-scale convolutional neural networks,” in *Proc. IEEE European Conf. on Comp. Vision*. Springer, 2016, pp. 154–169.
- [31] I. Goodfellow et al., “Generative adversarial nets,” in *Proc. Advances in Neural Information Proc. Systems*, 2014, pp. 2672–2680.
- [32] H. Zhang, V. Sindagi, and V. M. Patel, “Multi-scale single image dehazing using perceptual pyramid deep network,” in *Proc. IEEE Conf. Workshop on Comp. Vis. Patt. Recog.*, 2018, pp. 902–911.
- [33] R. Mondal, S. Santra, and B. Chanda, “Image dehazing by joint estimation of transmittance and airlight using bi-directional consistency loss minimized fcn,” in *Proc. IEEE Conf. Workshop on Comp. Vis. Patt. Recog.*, 2018, pp. 920–928.
- [34] H. Yang et al., “Image dehazing using bilinear composition loss function,” in *arXiv preprint arXiv:1710.00279*, 2017.
- [35] G. Huang et al., “Densely connected convolutional networks,” in *Proc. IEEE Conf. on Comp. Vis. Patt. Recog.*, 2017, pp. 4700–4708.
- [36] J. Kim, J. Kwon Lee, and K. Mu Lee, “Accurate image super-resolution using very deep convolutional networks,” in *Proc. IEEE Conf. on Comp. Vis. Patt. Recog.*, 2016, pp. 1646–1654.
- [37] S. Ioffe and C. Szegedy, “Batch normalization: Accelerating deep network training by reducing internal covariate shift,” *arXiv preprint arXiv:1502.03167*, 2015.
- [38] V. Monga, *Handbook of Convex Optimization Methods in Imaging Science*, Springer, 2017.
- [39] C. O. Ancuti et al., “Dense haze: A benchmark for image dehazing with dense-haze and haze-free images,” *arXiv preprint arXiv:1904.02904*, 2019.
- [40] C. Ancuti, C. O. Ancuti, and R. Timofte, “Ntire 2018 challenge on image dehazing: Methods and results,” in *Proc. IEEE Conf. Workshop on Comp. Vis. Patt. Recog.*, 2018, pp. 891–901.
- [41] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” *arXiv preprint arXiv:1412.6980*, 2014.
- [42] K. Zhang et al., “Learning deep cnn denoiser prior for image restoration,” in *Proc. IEEE Conf. on Comp. Vis. Patt. Recog.*, 2017, pp. 3929–3938.
- [43] Z. Wang et al., “Image quality assessment: from error visibility to structural similarity,” *IEEE Trans. on Image Proc.*, vol. 13, no. 4, pp. 600–612, 2004.
- [44] C. Ancuti et al., “I-haze: a dehazing benchmark with real hazy and haze-free indoor images,” in *International Conference on Advanced Concepts for Intelligent Vision Systems*. Springer, 2018, pp. 620–631.
- [45] C. O. Ancuti et al., “O-haze: a dehazing benchmark with real hazy and haze-free outdoor images,” in *Proc. IEEE Conf. Workshop on Comp. Vis. Patt. Recog.*, 2018, pp. 754–762.
- [46] K. He, J. Sun, and X. Tang, “Single image haze removal using dark channel prior,” in *Proc. IEEE Conf. on Comp. Vis. Patt. Recog.*, 2009.
- [47] Q. Zhu, J. Mai, and L. Shao, “A fast single image haze removal algorithm using color attenuation prior,” *IEEE Trans. on Image Proc.*, vol. 24, no. 11, pp. 3522–3533, 2015.