Learning Deeply Aggregated Alternating Minimization for General Inverse Problems

Hyungjoo Jung, *Member, IEEE*, Youngjung Kim, *Member, IEEE*, Dongbo Min, *Senior Member, IEEE*, Hyunsung Jang, *Member, IEEE*, Namkoo Ha, *Member, IEEE*, and Kwanghoon Sohn, *Senior Member, IEEE*

***Abstract*— Regularization-based image restoration is one of the most powerful tools in image processing and computer vision thanks to its flexibility for handling various inverse problems. However, designing an optimal regularization function still remains unsolved since natural images and related scene types have a complex structure. In this paper, we present a general and principled framework, called deeply aggregated alternating minimization (DeepAM). We design a convolutional neural network (CNN) to implicitly parameterize the regularizer of the alternating minimization (AM) algorithm. Contrary to the conventional AM algorithm based on a point-wise proximal mapping, the DeepAM projects intermediate estimate into a set of natural images via deep aggregation. Since the CNN is fully integrated into the AM procedure, all parameters can be jointly optimized through end-to-end training. These properties enable the DeepAM to converge with a small number of iterations, while maintaining an algorithmic simplicity. We show that the DeepAM outperforms state-of-the-art methods, including nonlocal-based methods, Plug-and-Play regularization, and recent data-driven approaches. The effectiveness of our framework is demonstrated in a variety of image restoration tasks: Guassian denoising, deraining, deblurring, super-resolution, color-guided depth upsampling, and RGB/NIR restoration.**

***Index Terms*— Regularization-based image restoration, joint restoration, convolutional neural network, alternating minimization, half-quadratic minimization, proximal mapping.**

I. Introduction

IMAGE restoration has been actively studied as an indispensable process in various image processing and computer vision applications. The goal of image restoration is to reconstruct a clean image from a degraded observation. The observed data is assumed to be related to the ideal image through a forward imaging model that accounts for noise [1], blurring [2], and/or sampling [3]. Consequently, image restoration is an inverse problem as it essentially amounts to inverting the forward model. This problem is highly ill-posed in the sense that there are more unknowns to estimate than measurements and the forward model is non-deterministic [4]. Simple modeling with observed data only produces an infinite number of feasible solutions. Thus, it is hopeless to choose among all possible ones without imposing an additional regularization that encodes our preference about good images. To that end, image restoration is usually formulated as an energy minimization problem consisting of data fidelity and explicit regularization terms [1], where it is most common to seek the image with lowest energy value. Additionally, joint image restoration leverages a guidance signal, captured from different imaging modalities, such as infrared, flash, and color images, as an external cue to regularize the restoration process. This approach is fundamentally helpful in various applications, including color-guided depth upsampling [5]–[7], cross-field noise removal [8], and infrared-guided dehazing [9].

Regularization-based image restoration involves minimization of non-smooth and non-convex energy functions for yielding high-quality restored results. Solving such problems typically requires a huge number of iterations, and thus an efficient optimization strategy is preferable. One of the most popular methods is alternating minimization (AM) algorithm [10] that introduces auxiliary variables. The energy function is decomposed into two sub-problems, which are related to data and regularization terms, respectively. Each sub-problem is relatively simple to minimize, allowing the use of a more sophisticated regularization. In the literature, the AM algorithm has been widely adopted with various regularization models, including image gradients [1], [2], nonlocal self-similarity [11], and group sparsity [12]. It is worth noting that these models are heavily engineered solutions to mimic the properties of natural images. Although the hyper-Laplacian of image gradients [2] reflects the long-tail statistics of natural images well, it is non-trivial to devise an optimal regularization function for a specific restoration problem. To address this issue, using a Gaussian radial basis function (RBF), several works [13], [14] have attempted to parameterize the regularization process (proximal mapping [13] or influence function [14]) of the AM algorithm. However, they assume

Manuscript received September 3, 2018; revised December 18, 2019; accepted July 9, 2020. Date of publication July 23, 2020; date of current version July 28, 2020. This work was supported by the Research and Development Program for Advanced Integrated-intelligence for Identification (AIID) through the National Research Foundation of Korea (NRF) funded by the Ministry of Science and ICT under Grant NRF-2018M3E3A1057289. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Xiaolin Wu. *(Corresponding author: Kwanghoon Sohn.)*

Hyungjoo Jung and Kwanghoon Sohn are with the School of Electrical and Electronic Engineering, Yonsei University, Seoul 120-749, South Korea (e-mail: coolguy0220@yonsei.ac.kr; khsohn@yonsei.ac.kr).

Youngjung Kim is with the Institute of Defense Advanced Technology Research, Agency for Defense Development, Daejeon 340-60, South Korea (e-mail: read12300@add.re.kr).

Dongbo Min is with the Department of Computer Science and Engineering, Ewha Womans University, Seoul 03-760, South Korea (e-mail: dbmin@ewha.ac.kr).

Hyunsung Jang is with the EO/IR R&D Laboratory, LIG Nex1 Company Ltd., Yongin 16911, South Korea, and also with the School of Electrical and Electronic Engineering, Yonsei University, Seoul 120-749, South Korea (e-mail: hyunsung.jang@yonsei.ac.kr).

Namkoo Ha is with the EO/IR R&D Laboratory, LIG Nex1 Company Ltd., Yongin 16911, South Korea (e-mail: hanamkoo369a@lignex1.com).

Digital Object Identifier 10.1109/TIP.2020.3010082

point-wise separable regularization functions, which is not flexible enough to describe complex structures of natural images.

Instead of carefully designing regularization functions, several approaches [15]–[17] have focused on the modular structure of the AM algorithm. It allows to replace the sub-problem related to the regularization term with off-the-shelf denoising methods, e.g., nonlocal mean [18] and BM3D [19], which serve as an implicit regularizer. Eventually, restored results are obtained by interleaving the sub-problem related to the data term and multiple denoising modules. Due to the heuristic nature, this type of algorithm is called the Plug-and-Play AM [15]–[17]. The Plug-and-Play AM has shown promising empirical results in a number of applications, such as deblurring, super-resolution, and inpainting. However, it requires manual parameter adjustments and lots of iterations to achieve satisfactory results.

In parallel, discriminative learning methods have been widely studied to solve the image restoration problems. The representative works include the anchored neighborhood regression [20], multi-layer perceptron [21], and convolutional neural network (CNN) [22]–[26]. With the help of large training data, these approaches learn a direct nonlinear function from degraded observation to ideal image. For example, low-resolution (LR) images are generated from high-resolution (HR) ones, and then the nonlinear function between the corresponding pairs can be learned using the CNNs [23], [25]. Similarly, the CNN can be used to solve image deblurring [24] or denoising [26]. While training of the CNN is very expensive, a high testing efficiency is achieved through parallel processing on GPU. The CNN-based methods have shown better performance than conventional regularization-based restoration methods. However, they have a limited capacity in adapting the observation models that characterize the degradation processes, i.e., the forward imaging models are not exploited and generally ignored. Furthermore, the CNN lacks regularization constraints over neighboring pixels, often bringing poor edge delineation and spurious regions. To overcome these limitations, several attempts have been made to combine the regularization-based approach with the CNNs. The works of [28]–[30] refine the results by integrating hand-crafted regularization model into top of the CNNs. They train the overall procedure in an end-to-end manner using the bilevel optimization technique. However, the bilevel optimization is solvable only when the energy function is convex and is twice differentiable [28]. In [31], the CNNs trained for image denoising are adopted in the Plug-and-Play AM algorithm. The CNN denoiser has advantages in exploiting large training data and leads to a more powerful regularizer, compared to conventional hand-designed denoisers [18], [19]. However, it is always pre-trained [31]–[33] and cannot be optimized jointly with other model parameters.

In this paper, we propose a generic method for image restoration that effectively uses the recent data-driven approach in the energy minimization framework, called *deeply aggregated alternating minimization* (DeepAM). Specifically, we design the CNN architectures to be operated as a proximal mapping for optimization algorithm. This naturally leads to the aggregated (or multivariate) mapping in the AM algorithm, showing better restoration performances than the conventional point-wise proximal mapping [2], [10], [13]. Since the CNNs are fully integrated into the AM algorithm, the whole networks can be jointly optimized in an end-to-end manner. Experimental results demonstrate that the DeepAM can achieve state-of-the-art performances on a number of restoration tasks. A short version of this work appeared in [34]. We extend our preliminary work in the following aspects.

- We provide an in-depth presentation of the DeepAM, and discuss its relations to other approaches.
- We extend [34] to handle various underdetermined inverse problems, i.e., deblurring and super-resolution by explicitly considering the observation models. For each task, we derive an efficient back-propagation rule for the end-to-end training.
- An intensive experimental study and comparison of the DeepAM with several state-of-the-art restoration methods are presented.

The remainder of this paper is organized as follows. Section II describes related works for image restoration. Section III provides some background and motivation of our work. We present the proposed method in Section IV. An extensive experimental evaluation is then provided in Section V. Finally, Section VI concludes this paper.

II. Related Works

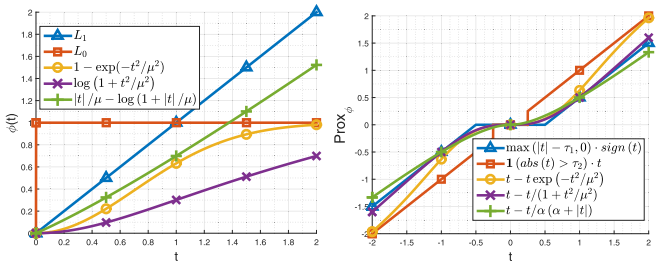
This section briefly reviews existing image restoration methods. Within various methodologies, we discuss three lines of research that are most relevant to ours: regularization- and CNN-based methods, and their hybrid approach.

A. Regularization-Based Image Restoration

The total variation (TV) [1] is the most popular regularization function that penalizes L_1-norm of image gradients. It forms a convex optimization problem which can be solved efficiently with a variety of algorithms, and guarantees a global optimality. The success of TV has prompted an in-depth study on regularization-based image restoration. Krishnan and Fergus [2] showed that the marginal distributions of real-world images have significantly heavier tails than L_1-norm, and adopted a hyper-Laplacian function. Other examples include Lorentzian function [35], total generalized variation (TGV) [36], and L_0-norm [37]. Motivated by the sparse property of images, sparsity priors have been widely incorporated as the prior knowledge of natural images. Moreover, Dong *et al.* [38] proposed to exploit the structural self-similarity of natural images with sparsity prior. Several approaches have attempted to learn regularization function from training data. Li *et al.* [39] proposed a hybrid parametric sparse model to learn the prior of HR images from training set and input LR images. Schmidt and Roth [13] proposed a cascade of shrinkage fields (CSF) using learned Gaussian RBF kernels. Similarly, Chen and Pock [14] modeled a nonlinear diffusion-reaction process [35] with parameterized linear filters and influence functions. Another closely related literature to ours is the Plug-and-Play regularization [15]–[17]. Venkatakrishnan *et al.* [15] showed

that the proximal mapping of alternating direction method of multiplier (ADMM) algorithm can be regarded as a single denoising step, and used an off-the-shelf denoiser [18], [19] for tomographic reconstruction. In the same spirit, Brifman *et al.* [16] recast the super-resolution problem into a series of denoising ones. Chan *et al.* proved the fixed point convergence of Plug-and-Play ADMM with the concept of bounded denoisers [17]. Although the Plug-and-Play ADMM has demonstrated the effectiveness in a number of applications, it requires tuning some parameters manually and lots of iterations to converge.

Joint restoration methods using a guidance image captured in different configurations have been also studied. In [5]–[7], HR RGB images were used to assist the regularization process of LR depth images. Shen *et al.* [8] proposed to use dark-flashed NIR images for the restoration of noisy RGB image. In [40], RGB image captured in dim light was restored using flash and non-flash pairs of the same scene.



(a) Regularization function (b) Proximal mapping

Fig. 1. (a) Illustrations of the regularization function Φ and (b) the corresponding proximal mapping. Intuitively, the main purpose of (b) is to pull small coefficients toward zero since they are assumed to be caused by noise. Instead of such handcrafted regularizers, we implicitly parameterize the regularization function using the deep aggregation, leading to a better restoration algorithm.

B. CNN-Based Methods

Inspired by the tremendous success of deep learning for high-level vision, the CNNs have been applied for low-level image restoration tasks. Direct nonlinear functions from degraded observation to clean image has been learned from the large-scale training data [22]–[27]. Note that their learning capability depends heavily on the choice of CNN architectures. For image super-resolution, Dong *et al.* [23] designed a shallow network that consists of three convolutional layers, and investigated its connection to traditional sparse coding-based methods. A similar architecture was used to solve image denoising and inpainting [22]. Xu *et al.* [24] cascaded 1D horizontal and vertical convolutional layers for deblurring, and avoided rapid parameter-size expansion. In general, a deeper and wider architecture does not necessarily produce better restoration results, due to the gradient vanishing problem. It was also noted that deeper networks would have higher training/testing errors than shallow one [23]. The works of [25], [26] resolved this problem by designing very deep CNNs with residual learning [41], and achieved state-of-the-art super-resolution and denoising performances. Bae *et al.* [27] indicated that the residual learning is a special case of manifold simplification. They further proposed to use a wavelet transform to simplify topological structures of degraded and clean image manifolds [27]. For joint restoration, Li *et al.* designed the CNN to selectively transfer salient structures that are consistent in both guidance and degraded images [42]. Hui *et al.* [43] extracted multi-scale guidance from RGB image for depth upsampling.

Despite their excellent performance, the aforementioned CNN-based methods do not ensure that the restored results are consistent with the observed images under the degradation processes. In contrast, the DeepAM explicitly takes into account the degradation models in a unified CNN framework.

C. Hybrid Approaches

The CNNs lack imposing the regularity constraint on adjacent similar pixels, often resulting in poor boundary localization and spurious regions. To deal with these issues, several approaches have attempted to integrate energy minimization models into the top of the CNNs [28]–[30]. Ranftl and Pock [28] defined unary and pairwise terms of Markov random fields (MRFs) using the outputs of the CNNs, and trained network parameters with the bilevel optimization. Similarly, the mean field approximation for fully connected conditional random fields was modeled as recurrent neural networks [44]. A non-local Huber regularization [30] and an anisotropic TGV [29] were combined with the CNNs for a high-quality depth upsampling. Note that these methods try to integrate hand-crafted regularization models into the top of CNNs. On the contrary, we use the CNN to parameterize the regularization process in the AM algorithm.

Recently, the work of [31] showed that CNN denoisers can be used for the Plug-and-Play AM algorithm. Bigdeli and Zwicker [32] trained denoising autoencoders, and adopted the squared magnitude of mean shift vectors as the regularizer. The resulting optimization problem is solved by gradient descent algorithm [32]. Meinhardt *et al.* [33] considered various convex optimization algorithms, where the proximal operator can be replaced by the CNN denoiser. These approaches can take advantages of both the regularization- and CNN-based restoration methods. However, The CNN denoiser and autoencoder used in [31]–[33] are pre-trained, which are not trained by end-to-end learning. Very recently, various works [45]–[48] have proposed to train the overall frameworks in an end-to-end manner. Yang *et al.* [45] designed effective unrolling architectures for compressive sensing MRI. Dianmond *et al.* [47] presented unrolled optimization with deep priors. In [46], they solved the problem of [49] (called OneNet) by training the overall framework of ADMM in an end-to-end manner. Dong *et al.* [48] optimized the reconstruction subproblem through a single step of gradient descent, which enables the whole parameters to be trained in an end-to-end manner. Different from [45]–[48], we derive efficient back-propagation rules with respect to various degradation models (including noise, blurring, and/or sampling). Furthermore, we apply our method into various restoration problems, including single and joint image restoration.

(a) Noisy input (b) TV [10] (c) CSF [13] (d) PnP [17] (e) Ours

Fig. 2. Examples of image deblurring: (a) input image, (b) TV [10], (c) CSF [13], (d) PnP [17], and (e) ours. Our aggregated mapping outperforms the traditional point-wise proximal mapping derived from TV [10] or learned Gaussian RGB [13]. The PnP [17] is implemented with the BM3D denoiser [19]. Although it is conceptually similar to our aggregation approach, they still relies on the handcrafted denoiser and the whole process cannot be optimized jointly.

III. Background and Motivation

The regularization-based image reconstruction typically formulates data fidelity term for degraded input and regularization term for output to be reconstructed. The output image is then computed by minimizing an objective function that balances these two terms. Assuming Gaussian additive noise, given an observed image $\mathbf{f}$ and a degradation matrix $\mathbf{A}$, we solve the corresponding optimization problem:

$$\arg\min_{\mathbf{u}} \frac{1}{2}\|\mathbf{Au}-\mathbf{f}\|_2^2 + \lambda\Phi\left(\mathbf{Gu}\right), \tag{1}$$

where λ is a balancing parameter. $\Phi(\cdot)$ is a regularization function that enforces the output image $\mathbf{u}$ to meet desired statistical properties. $\mathbf{G}$ is a feature extraction operator. The joint image restoration employs a guidance image $\mathbf{g}$, denoted as $\Phi(\mathbf{Gu}, \mathbf{g})$. The optimization problem of (1) can be solved using popular gradient descent methods [52], [53] or alternating minimization (AM) algorithms based on the variable splitting [10], [54]. In this paper, we focus on the additive[1] form of AM method [10], as its convergence is much faster than the gradient decent. We will show that unrolling of the small number of iterations is enough to achieve promising performances.

A. Alternating Minimization

By decoupling data and regularization terms with an auxiliary variable $\mathbf{v}$, the additive AM algorithm [10] reformulates (1) as the following constrained optimization problem:

$$\min_{\mathbf{u},\mathbf{v}} \frac{1}{2}\|\mathbf{Au}-\mathbf{f}\|_2^2 + \lambda\Phi\left(\mathbf{v}\right), \quad \text{subject to } \mathbf{Gu}=\mathbf{v}. \tag{2}$$

It clearly coincides with the unconstrained counterpart (1) in the feasible set $\{(\mathbf{u},\mathbf{v}) : \mathbf{Gu}=\mathbf{v}\}$. The constrained problem of (2) is then solved by the quadratic penalty technique [10], yielding the augmented objective function:

$$\min_{\mathbf{u},\mathbf{v}} \frac{1}{2}\|\mathbf{Au}-\mathbf{f}\|_2^2 + \lambda\Phi\left(\mathbf{v}\right) + \frac{\beta}{2}\|\mathbf{Gu}-\mathbf{v}\|_2^2, \tag{3}$$

[1]The multiplicative form [55] requires the inversion of linear matrix with a very large condition number, making it more difficult to perform the backpropagation.

where β is a penalty parameter. The AM algorithm performs the following steps iteratively with respect to $\mathbf{v}$ and $\mathbf{u}$:

$$\begin{aligned}
\mathbf{v}^{k+1} &= \arg\min_{\mathbf{v}} \frac{\beta^k}{2}\left\|\mathbf{Gu}^k-\mathbf{v}\right\|_2^2 + \lambda\Phi\left(\mathbf{v}\right),\\
\mathbf{u}^{k+1} &= \arg\min_{\mathbf{u}} \frac{1}{2}\|\mathbf{Au}-\mathbf{f}\|_2^2 + \frac{\beta^k}{2}\left\|\mathbf{Gu}-\mathbf{v}^{k+1}\right\|_2^2,\\
\beta^{k+1} &= \alpha\beta^k.
\end{aligned} \tag{4}$$

The penalty parameter β increases slowly by factor of $\alpha > 1$, i.e., continuation scheme. We denote by k the iteration index. When β is large enough, the variable $\mathbf{v}$ approaches $\mathbf{Gu}$, and thus the solution of (3) converges to that of (1). The rationale of this formulation is that each step of (4) may be much easier than the original unconstrained problem of (1). The AM algorithm [10] was widely used in hyper-Laplacian [2], Welsch's function [57], L_0-norm [37], and compound regularizer [56].

B. Motivation

Minimizing the first step in (4) varies depending on the choices of the regularization function Φ and β. This step can be regarded as the proximal mapping [12] of $\mathbf{u}^k$ associated with Φ. When Φ is the sum of L_1 norm or L_0 norm, it amounts to soft or hard thresholding operators (see Fig. 1 and [12] for various examples of this relation). The conventional AM method initializes β to a small constant and increases it gradually. For instance, soft thresholding operator, $v_i^{k+1} = \max\{|\mathbf{Gu}^k|_i - \frac{\lambda}{\beta^k}, 0\}\text{sign}\left(\mathbf{Gu}^k\right)_i$, shrinks large magnitudes of $\mathbf{Gu}^k$ when β is small. The high-frequency details of an image are recovered as β increases. Thus, using appropriate β is crucial for yielding high-quality restoration results. We argue that such mapping operators may not unveil the full potential of the regularization-based image restorations, since Φ and β are chosen manually. Furthermore, the mapping operator is performed for each pixel individually [10] in the feature space (e.g. filter and wavelet coefficients), disregarding spatial correlation with neighboring pixels.

We propose a new approach that learns the regularization function Φ and the penalty parameter β from the training dataset. Different from the point-wise proximal mapping based on the handcrafted regularizer, the proposed method learns and aggregates the mapping of $\mathbf{u}^k$ through CNNs.

IV. Proposed Method

This section first introduces the DeepAM for a single image restoration, and then extends the method to joint restoration tasks. In the following, the subscripts i denotes the location of a pixel in a vector form.

A. Deeply Aggregated Alternating Minimization

We begin with some intuition about why our learned and aggregated mapping is crucial to the AM algorithm. Traditionally, this mapping step has been applied in a point-wise manner, not to mention whether it is learned or not. With $\Phi(v) = \sum_i \phi(v_i)$, Schmidt and Roth [13] modeled the point-wise mapping function as Gaussian RBF kernels, and learned their mixture coefficients.[2] Contrarily, we do not presume any property of Φ. We instead train the aggregated mapping process associated with Φ and β by making use of the CNNs. In recent work [15], [17], the proximal mapping in (4) was replaced with off-the-shelf denoising algorithms such as non-local means [18] and BM3D [19]. This is conceptually similar to our DeepAM in that the spatial aggregation over neighboring pixels or patches is commonly performed in the denoising algorithms. However, they incorporate the denoising algorithm into the objective function of (1) in a plug-and-play (PnP) manner, and thus the whole model cannot be optimized jointly. Fig. 2 shows the deblurring examples of TV [10], CSF [13], PnP [17] and ours. Our model outperforms other methods using the point-wise mapping (Fig. 2(b) and (c)) or the PnP approach (Fig. 2(d)) (see the insets).

The DeepAM reformulates the original AM iterations in (4) as follows:

$$\mathbf{v}^{k+1} = \mathcal{D}^{\mathbf{u}}_{\mathrm{CNN}}(\mathbf{u}^k), \tag{5}$$

$$\mathbf{u}^{k+1} = \arg\min_{\mathbf{u}} \frac{1}{2}\|\mathbf{A}\mathbf{u} - \mathbf{f}\|_2^2 + \frac{\gamma^k}{2}\left\|\mathbf{u} - \mathbf{v}^{k+1}\right\|_2^2, \tag{6}$$

where the continuation parameter β is replaced with γ, which will be discriminatively learned through end-to-end training. λ is absorbed into $\mathcal{D}^{\mathbf{u}}_{\mathrm{CNN}}$ and fused with its parameters. $\mathbf{v}^{k+1}$ is estimated by deeply aggregating $\mathbf{u}^k$ through the CNN. Our reformulation in (5) and (6) allows to turn the optimization procedure in (1) into a cascaded CNN architecture, which can be learned by the standard back-propagation algorithm. It is worth noting that there are various ways to define $\mathbf{G}$ in (4). For instance, our preliminary work [34] employs $\mathbf{G} = \mathbf{D}$ which represents a gradient operator for both horizontal and vertical axes. This means the proximal mapping is performed in the gradient domain. When $\mathbf{G} = \mathbf{I}$, it maps an intermediate estimate into the image domain. We found through experiments that the performance variation of the two cases is very marginal. Please refer to Table I for performance analysis. In this work, we hence chose $\mathbf{G} = \mathbf{I}$ since it enables more efficient implementation of (6).

The solution of (6) satisfies the following linear system:

$$\mathbf{L}\mathbf{u}^{k+1} = \mathbf{A}^T\mathbf{f} + \gamma^k\mathbf{v}^{k+1}, \tag{7}$$

where $\mathbf{L} = \left(\gamma^k\mathbf{I} + \mathbf{A}^T\mathbf{A}\right)$ is a system matrix related to the observation matrix $\mathbf{A}$. In the following, we introduce how the $\mathbf{u}$-update step in (7) becomes the part of CNN as a reconstruction layer.

1) $\mathbf{A} = \mathbf{I}$ *(Denoising):* For image denoising, $\mathbf{v}^{k+1}$ is added back to the noisy input $\mathbf{f}$, and then re-scaled to estimate $\mathbf{u}^{k+1}$:

$$\mathbf{u}^{k+1} = \left(\mathbf{f} + \gamma^k\mathbf{v}^{k+1}\right) / \left(1 + \gamma^k\right). \tag{8}$$

This can be implemented efficiently using element-wise arithmetic operations. $\mathbf{u}^{k+1}$ is again fed into $\mathcal{D}^{\mathbf{u}}_{\mathrm{CNN}}$ for the next iteration. Such progressive denoising will show the improvement over the conventional methods [22], [26] that apply single forward network only once.

2) $\mathbf{A} = \mathbf{B}$ *(Non-Blind Deblurring):* The non-blind image deblurring is a special case when $\mathbf{A}$ is the circular convolution matrix, i.e., $\mathbf{B}\mathbf{u} = \mathbf{b} * \mathbf{u}$. In this case, since the system matrix $\mathbf{L}$ is diagonalizable, $\mathbf{u}^{k+1}$ can be computed by taking the fast Fourier transform (FFT) $\mathcal{F}(\cdot)$ on both sides of (7):

$$\mathbf{u}^{k+1} = \mathcal{F}^{-1}\left\{\frac{\mathcal{F}(\mathbf{A}^T)\mathcal{F}(\mathbf{f}) + \gamma^k\mathcal{F}(\mathbf{v}^{k+1})}{\gamma^k\mathcal{F}(\mathbf{I}) + |\mathcal{F}(\mathbf{A})|^2}\right\}, \tag{9}$$

where the multiplication/division are all component-wise operations. Once the constant quantities are computed, updating $\mathbf{u}$ requires two FFT calls.

3) $\mathbf{A} = \mathbf{S}\mathbf{B}$ *(Super-Resolution):* Image super-resolution can be described by an anti-aliasing $\mathbf{B}$ and a sub-sampling $\mathbf{S}$ matrices. The system matrix $\mathbf{L} = \left(\gamma^k\mathbf{I} + \mathbf{A}^T\mathbf{A}\right)$ is no longer diagonalizable by the FFT. One can use the inner conjugate gradient decent to solve (7), but it is computationally expensive and is not suitable for parallel implementation. We instead adopt the polyphase decomposition method proposed in [17]. The $\mathbf{u}$-update step for image super-resolution can be rewritten using the Woodbury matrix identity [58]:

$$\mathbf{u}^{k+1} = \frac{\mathbf{z}}{\gamma^k} - \frac{1}{\gamma^k}\mathbf{A}^T\left(\gamma^k\mathbf{I} + \mathbf{A}\mathbf{A}^T\right)^{-1}\mathbf{A}\mathbf{z}, \tag{10}$$

where $\mathbf{z} = \mathbf{A}^T\mathbf{f} + \gamma^k\mathbf{v}^{k+1}$ and $\mathbf{A}\mathbf{A}^T = \mathbf{S}\mathbf{B}\mathbf{B}^T\mathbf{S}^T$. We now need to compute the inversion of $\left(\gamma^k\mathbf{I} + \mathbf{A}\mathbf{A}^T\right)$. Note that since $\mathbf{S}$ is a sub-sampling matrix, $\mathbf{S}^T$ corresponds to a up-sampling matrix. The polyphase decomposition of $\mathbf{A}\mathbf{A}^T$ illustrates the fact that this operation is equivalent to applying the 0-th polyphase component of the $\mathbf{B}\mathbf{B}^T$, since a time delay between up-sampling and down-sampling operators leads to a zero response (see [17] for more details). Thus, denoting $\bar{B}_0$ as the 0-th polyphase component of the $\mathbf{B}\mathbf{B}^T$, (10) can be implemented with the FFT:

$$\mathbf{u}^{k+1} = \frac{\mathbf{z}}{\gamma^k} - \frac{1}{\gamma^k}\mathbf{A}^T\left(\mathcal{F}^{-1}\left\{\frac{\mathcal{F}(\mathbf{A}\mathbf{z})}{\gamma^k\mathcal{F}(\mathbf{I}) + |\mathcal{F}(B_0)|}\right\}\right). \tag{11}$$

The closed-form solution of (11) is exact under the circular boundary condition.

B. Extension to Joint Image Restoration

This section extends the proposed method to joint image restoration tasks. The basic idea of joint restoration is to provide structural guidance, assuming structural correlation between different kinds of feature maps, e.g., depth/RGB

[2] When $\Phi(v) = \sum_i \phi(v_i)$, the first step in (4) is separable with respect to each v_i. It can be thus modeled by point-wise operation.

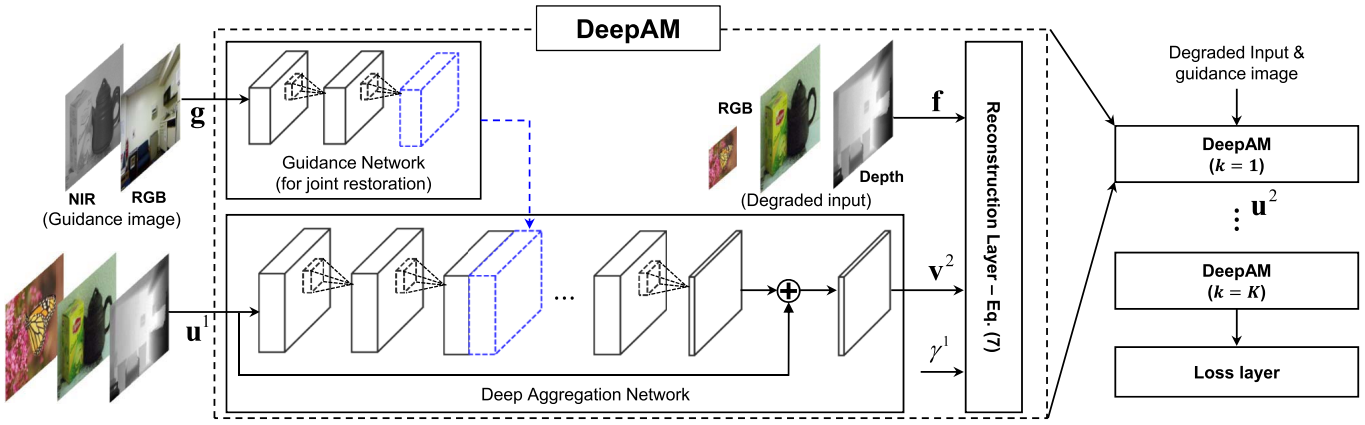


Fig. 3. One iteration of our model consists of three major components: deep aggregation network, guidance, and reconstruction layer. For joint image restoration, the guidance network takes $\mathbf{g}$ as input, and extracts feature maps which are then combined with intermediate features of the deep aggregation network. All of these sub-networks are cascaded by iterating (5) and (6), and the final output is then entered into the loss layer.

and NIR/RGB. Such a constraint has been imposed on the conventional proximal mapping by considering structures of both input and guidance images [57], [59]. Similarly, one can modify the deeply aggregated mapping of (5) as follows:

$$\mathbf{v}^{k+1} = \mathcal{D}^{\mathbf{u}}_{\text{CNN}}(\mathbf{u}^k \otimes \mathbf{g}), \tag{12}$$

where $\otimes$ denotes a concatenation operator. However, we observed that such early concatenation is less effective due to homogeneous attributes of $\mathbf{g}$ and $\mathbf{u}$. This coincides with the observation in the literature of multi-modal feature learning [60], [61]. Thus, we adopt the halfway concatenation strategy [60] by introducing the sub-network $\mathcal{D}^{\mathbf{g}}_{\text{CNN}}$ for an effective representation of $\mathbf{g}$ and then combining it with intermediate features of $\mathcal{D}^{\mathbf{u}}_{\text{CNN}}(\mathbf{u}^k)$.

C. Network Architecture

One iteration of the proposed DeepAM consists of three major parts: deep aggregation network, guidance network (for joint restoration), and reconstruction layer, as shown in Fig. 3. The deep aggregation network consists of 15 convolutional layers with small 3×3 filters (a receptive field is of 31×31). Each layer generates 32 feature maps with the ReLU nonlinearity, except for the last convolutional layer. The batch normalization is used to reduce the internal covariate shift [62]. A skip shortcut from the input ($\mathbf{u}^k$) to the output ($\mathbf{v}^{k+1}$) is added to the deep aggregation network (see Fig. 3). For joint image restoration, the guidance network consists of 3 convolutional layers, where the filters operate on 3×3 spatial region. It takes the guidance image $\mathbf{g}$ as input, and extracts feature maps which are then concatenated with the third convolutional layer of the deep aggregation network. The reconstruction layer is implemented according to (8), (9), or (11) for each application. All of these are cascaded by iterating (5) and (6), and the final output ($\mathbf{u}^K$) is then entered into the loss layer.

D. Training

The DeepAM is learned with standard back-propagation algorithm in an end-to-end manner. We do not require any approximation or complicate bi-level formulation [29]. Given M training image pairs $\{\mathbf{f}^{(p)}, \mathbf{g}^{(p)}, \mathbf{t}^{(p)}\}_{p=1}^{M}$, we train the network by minimizing the L_2-loss function:

$$\mathcal{L} = \frac{1}{M} \sum_p \left\| \mathbf{u}^{(p)} - \mathbf{t}^{(p)} \right\|_2^2. \tag{13}$$

$\mathbf{u}^{(p)}$ and $\mathbf{t}^{(p)}$ denote the output of the last reconstruction layer, i.e., $\mathbf{u}^K$ for p-th training sample and the ground-truth image, respectively. The Adam solver is used with default setting to minimize the loss function of (13). The derivative for the back-propagation is obtained as follows:

$$\frac{\partial \mathcal{L}^{(p)}}{\partial \mathbf{u}^{(p)}} \propto \left(\mathbf{u}^{(p)} - \mathbf{t}^{(p)} \right). \tag{14}$$

At each iteration k, we need the derivatives of the loss $\mathcal{L}^{(p)}$ with respect to $\mathbf{v}^{(p)}$ and $\gamma^{(p)}$ through the reconstruction layer. Using the chain rule of differentiation, we have the following expressions. From here on, the iteration and sample indexes are omitted to simplify notations.

$$\frac{\partial \mathcal{L}}{\partial \mathbf{v}} = \frac{\partial \mathbf{u}}{\partial \mathbf{v}} \frac{\partial \mathcal{L}}{\partial \mathbf{u}}, \quad \frac{\partial \mathcal{L}}{\partial \gamma} = \frac{\partial \mathbf{u}}{\partial \gamma} \frac{\partial \mathcal{L}}{\partial \mathbf{u}}. \tag{15}$$

$\frac{\partial \mathcal{L}}{\partial \mathbf{v}}$ and $\frac{\partial \mathcal{L}}{\partial \gamma}$ will have different differential forms in denominator layout according to the observation matrix $\mathbf{A}$.

1) $\mathbf{A} = \mathbf{I}$*:* From (8), it is straightforward to see that $\frac{\partial \mathcal{L}}{\partial \mathbf{v}}$ is directly proportional to $\frac{\partial \mathcal{L}}{\partial \mathbf{u}}$:

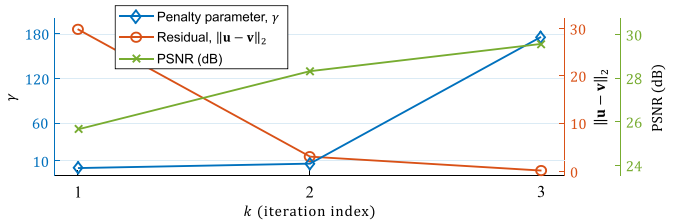
$$\frac{\partial \mathcal{L}}{\partial \mathbf{v}} = \frac{\gamma}{1+\gamma} \frac{\partial \mathcal{L}}{\partial \mathbf{u}}. \tag{16}$$

$\frac{\partial \mathcal{L}}{\partial \gamma}$ can be obtained with the product rule of differentiation:

$$\frac{\partial \mathcal{L}}{\partial \gamma} = \frac{1}{1+\gamma} (\mathbf{v} - \mathbf{u})^T \frac{\partial \mathcal{L}}{\partial \mathbf{u}}. \tag{17}$$

In the case of denoising, the back-propagation through reconstruction layer is performed using simple arithmetic operations (multiplication and inner product).

(a) $k = 1$ (b) $k = 2$ (c) $k = 3$



(d) Learned γ, sum of pixel difference between $\mathbf{u}$ and $\mathbf{v}$, and PSNRs

Fig. 4. Denoising examples obtained by our DeepAM (trained with $K = 3$ iterations jointly). See the text for details.

2) $\mathbf{A} = \mathbf{B}$*:* Differentiating both side of (7) with respect to $\mathbf{v}$, we have:

$$\frac{\partial \mathcal{L}}{\partial \mathbf{v}} = \gamma \left(\gamma \mathbf{I} + \mathbf{A}^T \mathbf{A}\right)^{-1} \frac{\partial \mathcal{L}}{\partial \mathbf{u}} = \gamma \mathbf{L}^{-1} \frac{\partial L}{\partial \mathbf{u}}. \tag{18}$$

Thus, $\frac{\partial \mathcal{L}}{\partial \mathbf{v}}$ for deblurring is determined by solving the linear system of (18). Since γ only appears in the diagonal component of $\mathbf{L}$, we obtain the following expression for $\frac{\partial \mathcal{L}}{\partial \gamma}$:

$$\frac{\partial \mathcal{L}}{\partial \gamma} = (\mathbf{v} - \mathbf{u})^T \left(\mathbf{L}^{-1} \frac{\partial \mathcal{L}}{\partial \mathbf{u}}\right). \tag{19}$$

The detailed derivation of (19) is available in Appendix A. For image deblurring, the backward step through the reconstruction layer can be performed efficiently using the FFT.

3) $\mathbf{A} = \mathbf{SB}$*:* Letting $\mathbf{K} = \mathbf{A}^T \left(\gamma \mathbf{I} + \mathbf{A}\mathbf{A}^T\right)^{-1} \mathbf{A}$ and differentiating both side of (10), $\frac{\partial \mathcal{L}}{\partial \mathbf{v}}$ is derived as follows:

$$\frac{\partial \mathcal{L}}{\partial \mathbf{v}} = (\mathbf{I} - \mathbf{K})^T \frac{\partial \mathcal{L}}{\partial \mathbf{u}}. \tag{20}$$

We then multiply γ to both side of (10), and take the partial derivative with respect to γ:

$$\mathbf{u}^T + \gamma \frac{\partial \mathbf{u}}{\partial \gamma} = -\mathbf{z}^T \left(\frac{\partial \mathbf{K}}{\partial \gamma}\right)^T + \frac{\partial \mathbf{z}}{\partial \gamma} (\mathbf{I} - \mathbf{K})^T, \tag{21}$$

where we recall that $\mathbf{z} = \mathbf{A}^T \mathbf{f} + \gamma \mathbf{v}$. After some calculation, with the identity $\partial \mathbf{A}^{-1} / \partial \gamma = -\mathbf{A}^{-1} (\partial \mathbf{A} / \partial \gamma) \mathbf{A}^{-1}$ [51], $\frac{\partial \mathcal{L}}{\partial \gamma}$ is obtained as:

$$\frac{\partial \mathcal{L}}{\partial \gamma} = \frac{1}{\gamma} \left\{\mathbf{z}^T \mathbf{A}^T \mathbf{P}^2 \mathbf{A} + \mathbf{v}^T (\mathbf{I} - \mathbf{K}) - \mathbf{u}^T\right\} \frac{\partial \mathcal{L}}{\partial \mathbf{u}}, \tag{22}$$

where $\mathbf{P} = \left(\gamma \mathbf{I} + \mathbf{A}\mathbf{A}^T\right)^{-1}$. More details about the derivation of (22) is available in Appendix B. Note that we can implement (20) and (22) in closed-form using the polyphase decomposition, as in Section IV-A.

Figure 4 shows the denoising results of our method. Here, it is jointly trained with three passes of DeepAM, i.e., only the output of last reconstruction layer passes through the loss layer. The input image is corrupted by Gaussian noise with standard deviation $\sigma = 25$. We see that as the iteration proceeds, the high-quality restoration results are produced (Figs. 4(a)-(c)). We report the learned penalty parameter γ at each iteration in Fig. 4(d). The conventional AM method sets γ as a small constant and increases it during iterations. Interestingly, the DeepAM shows very similar behavior,[3] but outperforms the existing methods thanks to the aggregated mapping through the CNNs, as will be validated in experiments. Furthermore, the DeepAM automatically satisfies the equality constraint in (2). As shown in the red line of Fig. 4(d), $\|\mathbf{u} - \mathbf{v}\|_2$ is almost zero at the last iteration. The green line of Fig. 4(d) denotes the PSNRs of $\mathbf{u}^k$ (25.85, 28.33, and 29.44 dBs for each iteration).

V. Experiments

The DeepAM is trained end-to-end for 30 epoches, given the degraded observation and the groud-truth. We use an initial learning rate of 10^{-3} which is kept constant for the first 10 epochs. After that it is halved every 5 epoches until the end. The exponential decay rates for the first and the second moments are set to 0.9 and 0.999, respectively. The MatConvNet[4] with 12GB NVIDIA TITAN GPU is used for network construction and training. We do not perform any pre-training, that is, the networks are initialized randomly using Gaussian distributions. In the sequel, we call DeepAM$^{(K)}$ the method trained through a cascade of K DeepAM iterations. We evaluate the DeepAM on six restoration tasks, including denoising, deraining, deblurring, super-resolution, color-guided depth upsampling, and RGB/NIR restoration. The last two are joint restoration tasks, incorporating an additional guidance image captured in different sensors.

A. Single Image Denoising

To generate the training data for denoising experiments, we follow [13] to use 400 images from the Berkeley segmentation (BSD) dataset [69]. These are strictly separated from all test images. We extract 2×10^5 image patches of size 64×64, and add Gaussian noise with $\sigma = 15$, 25, and 50 to synthetically generate noisy patches. The data augmentation is performed on the fly. We flip and rotate the input patches with a 25% chance. The total number of iterations is set to $K = 3$ as the performance converges after 3 iterations (refer to Table II). We compared against a variety of recent state-of-the-art techniques, including BM3D [19], WNNM [63], EPLL [64], CSF [13], TRD [14], DnCNN [26], WDnCNN [27], and DPDNN [48]. The first two methods are based on the non-local regularization, and the others are learning-based approaches. Especially, DnCNN [26] and WDnCNN [27] use single forward CNN with residual learning of image and wavelet domain, respectively. DnCNN [26] consists of 17 convolutional layers and 64 intermediate feature maps. WDnCNN [27] has 20 convolutional layers with 320 intermediate feature maps. Set12 [19] and BSD68 [69] are used as the test set.

[3] We do not impose any constraints that γ should increase during successive steps. γ for each iteration is initialized to 1.

[4] http://www.vlfeat.org/matconvnet/

(a) Noisy input (b) BM3D [19] (c) EPLL [64] (d) WNNM [63] (e) TRD [14] (f) DnCNN [26] (g) DeepAM$^{(3)}$

Fig. 5. Denoising examples with $\sigma = 50$: (a) noisy input, (b) BM3D [19], (c) EPLL [64], (d) WNNM [63], (e) TRD [14], (f) DnCNN [26], and (g) DeepAM$^{(3)}$. The input images are from the BSD68 [69]. PSNR/SSIM values are included in upper-right.

TABLE I

THE PSNR RESULTS ON 12 IMAGES WITH $\sigma = 15$, 25, AND 50. THE CSF [13] AND TRD [14] RUN 5 STAGES WITH 7×7 KERNELS

	C.man	House	Pepp.	Starf.	Monar.	Airpl.	Parrot	Lena	Barb.	Boat	Man	Couple	Average
Noise Level	$\sigma = 15$												
BM3D [19]	31.91	34.93	32.69	31.14	31.85	31.07	31.37	34.26	33.10	32.13	31.92	32.10	32.372
WNNM [63]	32.17	35.13	32.99	31.82	32.71	31.39	31.62	34.27	**33.60**	32.27	32.11	32.17	32.696
CSF [13]	31.95	34.39	32.85	31.55	32.33	31.33	31.37	34.06	31.92	32.01	32.08	31.98	32.318
TRD [14]	32.19	34.55	33.03	31.76	32.57	31.47	31.63	34.25	32.14	32.15	32.24	32.11	32.510
DnCNN [26]	32.62	35.00	33.29	32.23	33.10	31.70	31.84	34.63	32.65	32.42	32.47	32.47	32.920
DPDNN [48]	32.44	**35.40**	33.19	32.08	**33.33**	**31.78**	31.48	**34.80**	32.84	**32.55**	**32.53**	32.51	32.911
DeepAM$^{(3)}_{\mathbf{G}=\mathbf{D}}$	32.58	35.26	33.31	32.21	33.06	31.75	31.88	34.66	32.91	32.45	32.46	32.51	32.880
DeepAM$^{(3)}$	**32.70**	35.20	**33.35**	**32.26**	33.09	31.74	**31.89**	34.70	32.83	32.50	32.48	**32.59**	**32.944**
Noise Level	$\sigma = 25$												
BM3D [19]	29.45	32.85	30.16	28.56	29.25	28.42	28.93	32.07	30.71	29.90	29.61	29.71	29.969
WNNM [63]	29.64	33.23	30.42	29.03	29.84	28.69	29.15	32.24	**31.24**	30.03	29.76	29.82	30.257
CSF [13]	29.48	32.39	30.32	28.80	29.62	28.72	28.90	31.79	29.03	29.76	29.71	29.53	29.837
TRD [14]	29.72	32.53	30.57	29.02	29.85	28.88	29.18	32.00	29.41	29.91	29.87	29.71	30.055
DnCNN [26]	30.18	33.06	30.87	29.41	30.28	29.13	29.43	32.44	30.00	30.21	30.10	30.12	30.436
DPDNN [48]	30.12	**33.54**	30.90	29.43	**30.31**	29.14	29.28	**32.69**	30.30	**30.34**	**30.15**	30.24	30.537
DeepAM$^{(3)}_{\mathbf{G}=\mathbf{D}}$	30.20	33.32	30.93	29.46	30.24	29.16	29.44	32.55	30.37	30.24	30.11	32.23	32.520
DeepAM$^{(3)}$	**30.30**	33.43	**30.94**	**29.50**	30.26	**29.16**	**29.44**	32.63	30.29	30.30	30.11	**30.31**	**30.555**
Noise Level	$\sigma = 50$												
BM3D [19]	26.13	29.69	26.68	25.04	25.82	25.10	25.90	29.05	27.23	26.78	26.81	26.46	26.730
WNNM [63]	26.42	30.33	26.91	25.43	26.32	25.42	26.09	29.25	**27.79**	26.97	26.94	26.64	27.040
TRD [14]	26.62	29.48	27.10	25.42	26.31	25.59	26.16	28.93	25.70	26.94	26.98	26.50	26.810
DnCNN [26]	27.00	30.02	27.29	25.70	26.77	25.87	26.48	29.37	26.23	27.19	27.24	26.90	27.170
DPDNN [48]	27.12	**31.04**	27.44	**25.95**	**27.00**	**25.97**	26.42	**29.85**	27.21	**27.42**	**27.32**	**27.23**	**27.498**
DeepAM$^{(3)}_{\mathbf{G}=\mathbf{D}}$	27.06	30.40	**27.51**	25.82	26.91	25.94	26.49	29.59	26.86	27.30	27.29	27.10	27.355
DeepAM$^{(3)}$	**27.15**	30.60	27.48	25.82	26.95	25.93	**26.50**	29.64	26.75	27.35	27.28	27.17	27.383

Table I shows the peak signal-to-noise ratio (PSNR) on the Set12 [19] images. The best results for each image are highlighted in bold. The DeepAM$^{(3)}_{\mathbf{G}=\mathbf{D}}$ indicates our preliminary version [34]. We trained DeepAM$^{(3)}_{\mathbf{G}=\mathbf{D}}$ again with the same network architecture in Section IV-C. We observed that the feature extraction ($\mathbf{G} = \mathbf{D}$) has little effect on the denoising performance. We could find that our deep aggregation used in the proximal mapping outperforms the point-wise mapping of

TABLE II

AVERAGE PSNR/SSIM ON BSD68 FOR IMAGE DENOISING WITH $\sigma = 15, 25$, AND 50. NOTE THAT, IN THIS EXPERIMENT, WE FORCE THE DEEPAM$^{(1)\text{ to }(3)}$ TO HAVE THE SAME NUMBER OF PARAMETERS

	PSNR / SSIM								
	BM3D [19]	EPLL [64]	CSF [13]	TRD [14]	DnCNN [26]	WDnCNN [27]	DeepAM$^{(1)}$	DeepAM$^{(2)}$	DeepAM$^{(3)}$
15	31.08 / 0.872	31.19 / 0.883	31.24 / 0.873	31.42 / 0.882	31.74 / 0.891	**31.84 / 0.893**	31.58 / 0.885	31.77 / 0.889	31.81 / 0.892
25	28.57 / 0.802	28.68 / 0.812	28.73 / 0.803	28.91 / 0.816	29.23 / 0.828	–	29.12 / 0.813	29.27 / 0.830	**29.31 / 0.832**
50	25.62 / 0.687	25.68 / 0.688	–	25.96 / 0.702	26.24 / 0.719	**26.38 / 0.728**	26.16 / 0.716	26.25 / 0.720	26.33 / 0.723

(a) Rainy image (b) DetailNet [68] (c) ID-CGAN [70] (d) DeepAM$^{(3)}$

Fig. 6. Results of rain removal on real rainy images: (a) rainy image, (b) DetailNet [68], (c) ID-CGAN [70], and DeepAM$^{(3)}$.

CSF [13] by 0.62 $\sim$ 0.71 dB. This table also demonstrates the advantages of the progressive denoising of DeepAM$^{(3)}$ over DnCNN [26] which applies the single forward network only once. Note that our method and DnCNN [26] produces 64 and 32 intermediate feature maps. Although three passes of forward networks are used, DeepAM$^{(3)}$ has 1.5 times fewer parameters than DnCNN [26]. DPDNN [48] has shown comparable results with our method. In [48], they obtained an inexact solution for (6) with a single step of gradient descent. As a result, our method requires less iterations of AM algorithm (3 iterations) than [48] (6 iterations). The performance gain becomes larger for higher noise levels. Learning-based methods tend to have better performance than hand-crafted models. However, we observe that WNNM [63] based on the non-local regularization works very well on images that are dominated by repetitive textures, e.g., 'House' and 'Barbara'. The non-local self-similarity is a powerful prior on regular and repetitive texture, but it may lead to inferior results on irregular regions.

Fig. 5 shows denoising examples, sampled from the BSD68 [69]. The input image is contaminated by Gaussian noise with $\sigma = 50$. One can see that the results of BM3D [19], EPLL [64], and WNNM [63] contain many visual artifacts, and tend to oversmooth image structures. TRD [14] and DnCNN show better visual quality, but produce spurious details in homogeneous areas. In contrast, the DeepAM$^{(3)}$ shows more promising result, and visually outperforms the other methods. It successfully suppresses noises and visual artifacts while reproducing sharper edges and details.

Table II summarizes an objective evaluation by measuring average PSNR and structural similarity index metric (SSIM) [65] on the BSD68 [69] test images. As expected, the DeepAM$^{(3)}$ achieves a significant improvement over the non-local based method [19], [63] as well as the recent learning-based approaches [13], [14], [26], [64]. Even though WDnCNN [27] achieves higher PSNR and SSIM than the DeepAM$^{(3)}$, the performance gain is marginal considering the number of network parameters. WDnCNN [27] uses about 30 times larger number of parameters than the DeepAM$^{(3)}$. We also analyzed our results with different choice of $K = 1$, 2, and 3. For a fair comparison, we force the DeepAM$^{(1)\text{ to }(3)}$ to have the same number of parameters by adjusting the number of feature maps. The numbers of intermediate feature maps for DeepAM$^{(1)\text{ to }(3)}$ are thus 64, 48, and 32, respectively. We can see that our progressive scheme via unrolling is indeed beneficial, and the improvement is saturated after 3 iterations.

B. Rain Streak Removal

Removing rain streaks in outdoor scenes is important for many computer vision and photography applications. The rain streak is associated with visibility decrease that causes image features to be less distinctive and makes many vision systems easily fail. Given a rainy image $\mathbf{f}$, rain streak removal aims to decompose the image as the summation of two components: $\mathbf{f} = \mathbf{u} + \mathbf{r}$, where $\mathbf{r}$ denotes a rain streak layer. We directly apply our denoising model to recover a rain-free background image $\mathbf{u}$ from the rainy image $\mathbf{f}$. We used

(a) Blurry image (b) CSF [13] (c) DAE [32] (d) PnP_CNN [31] (e) DeepAM$^{(3)}$

Fig. 7. Deblurring results on real blurry images: (a) blurry image, (b) CSF [13], (c) DAE [32], (d) PnP_CNN [31], and (e) DeepAM$^{(3)}$. The blur kernels are estimated by [73], weher are shown in the upper-left of (a).

TABLE III

AVERAGE PSRN ON 100 SYNTHETIC TEST SET FROM [70] FOR RAIN STREAK REMOVAL

Method	GMM [67]	DetailNet [68]	ID-CGAN [70]	DeepAM$^{(3)}$
Avg. PSNR	22.27	23.48	22.73	**25.24**

synthetic rainy images from [70] to generate training data. The dataset contains a total of 700 clean images for training, where 500 images are randomly sampled from the UCID dataset [66] and 200 images are chosen from the BSD dataset [69]. The rain streaks are synthesized using the Photoshop with different rain intensities and orientations. We randomly sampled 2×10^5 patch pairs of size 64×64 from the training set, and trained the DeepAM$^{(3)}$. We compared our method with several leading approaches, including GMM [67], DetailNet [68] and ID-CGAN [70]. The last two methods are based on the CNN equipped with the off-the-shelf image decomposition [68] and conditional adversarial learning [70]. For fair comparison, the DetailNet [68] was retrained on the same training set using the source code provided by the author. The test set consists of 100 images that are randomly chosen from the UCID [66] and the BSD [69] datasets, but are not included in training set.

We report the quantitative measures using PSNR in Table III. This table clearly shows that the DeepAM$^{(3)}$ achieves superior quantitative performance with a large margin. The results on two real-world rainy images are shown in Fig. 6. For better visualization, we show zoomed versions of the two regions in the insets. The top example contains light rain streaks, and the bottom one is with heavy rain. The DetailNet [68] fails to remove heavy rain streaks as shown in Fig. 6(b). The ID-CGAN [70] tries to make the prediction indistinguishable from natural clean images using the adversarial loss, but produces obvious visual artifacts and over-smoothed results (see Fig. 6(c)). Consistently, the DeepAM$^{(3)}$ outperforms the other methods, and is successful in removing the majority of rain streaks.

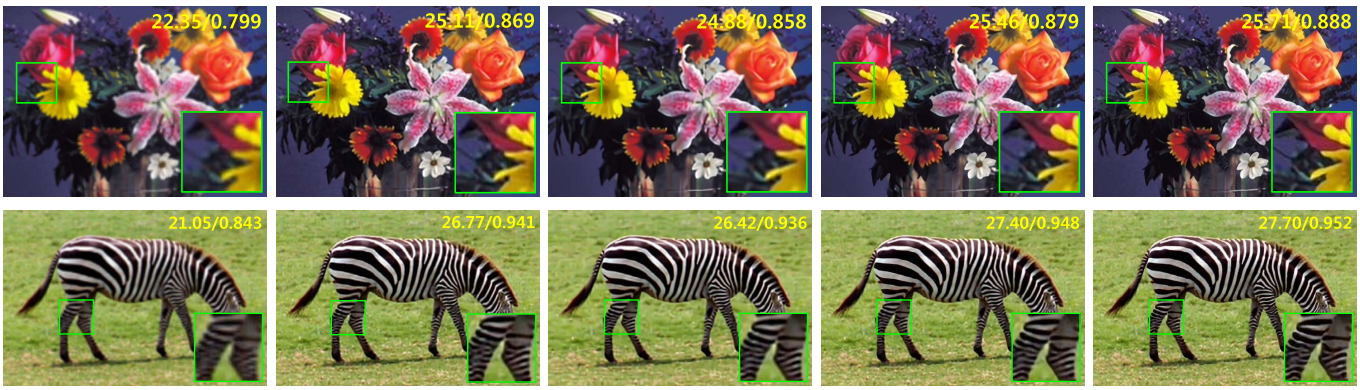
C. Non-Blind Image Deblurring

We apply our DeepAM$^{(3)}$ into non-blind image deblurring with estimated blur kernels. We used 400 clean images from BSD dataset [69]. Two types of blur kernels are considered: 25×25 Gaussian blur kernel of standard deviation 1.6 and motion kernels from [71]. The motion kernels consist of 8 examples with different size ranged from 13 to 27. We first obtained 3600 blurry training images by convolving clean images with the blur kernels, followed by adding small Gaussian noise with $\sigma = 2.55$. Then, blur kernels are estimated from the degraded images using [72] and [73]. The DeepAM$^{(3)}$ was trained by 2×10^5 patch pairs from the clean/blurry image pairs with the estimated blur kernels. We compared our model with three non-blind image deblurring methods, i.e., CSF [13], DAE [32] and PnP_CNN [31], which are highly relevant to ours. The first method learns the point-wise proximal mapping, and the others replace the mapping with the CNN denoiser as a plug-and-play prior. The deblurred results of color images are obtained from the luminance channel only except for DAE [32] designed for color image deblurring. For a fair comparison, we trained CSF again with our training dataset using a code from the author's website.[5]

Table IV shows PSNR values of the competing methods on the classic 6 gray images. Test images were also generated in a manner similar to that of training images by convolving clean images with three blur kernels (See Table IV), followed by adding Gaussian noise with $\sigma = 2.55$. We applied the non-blind deblurring algorithms with both ground truth blur kernels and estimated kernels from [72]. For both cases, DeepAM$^{(3)}$ outperforms the competing methods with large margin. PnP_CNN [31] shows limited performance in the case of inaccurate blur kernels.

Fig. 7 shows the results on the naturally blurred images. The motion kernels are estimated from [73], which are depicted in the upper-left of Fig. 7(a). Each kernel size is 35×35 and

[5]https://github.com/uschmidt83/shrinkage-fields



(a) Zoomed LR image (b) NCSR [74] (c) PnP [17] (d) PnP_CNN [31] (e) DeepAM$^{(3)}$

Fig. 8. Example of image super-resolution: (a) zoomed LR image, (b) NCSR [74], (c) PnP [17], (d) PnP_CNN [31], and (e) DeepAM$^{(3)}$. The LR image is obtained by Gaussian blur kernel, followed by downsampling with the scale factor 3. PSNR/SSIM values are included in upper-right.

TABLE IV

AVERAGE PSNR ON 6 TEST IMAGES FOR NONBLIND IMAGE DEBLURRING. TWO MOTION KERNELS ARE CONSIDERED FROM [71]

Method	Kernel	C.man	House	Lena	Monar.	Leaves	Parrots	Average
25×25 Gaussian blur kernel with standard deviation 1.6								
CSF [13]	GT	25.90	32.49	32.67	26.82	27.25	29.00	29.02
PnP_CNN [31]		**27.73**	33.46	30.81	29.14	29.11	30.46	30.12
DeepAM$^{(3)}$		27.41	**33.77**	**32.33**	**29.60**	**29.66**	**30.92**	**30.62**
CSF [13]	[72]	24.52	25.32	27.56	26.94	24.89	24.32	25.59
PnP_CNN [31]		25.61	25.14	28.91	28.09	26.08	25.24	26.51
DeepAM$^{(3)}$		**25.96**	**26.92**	**30.11**	**28.76**	**27.00**	**26.01**	**27.46**
17×17 Motion kernel 1								
CSF [13]	GT	28.83	32.66	32.77	25.76	25.39	31.26	29.46
PnP_CNN [31]		31.29	34.82	33.35	31.39	32.54	33.71	32.85
DeepAM$^{(3)}$		**32.47**	**35.35**	**35.67**	**32.53**	**33.45**	**34.74**	**34.04**
CSF [13]	[72]	21.55	23.84	28.87	27.02	27.51	25.94	25.79
PnP_CNN [31]		23.25	21.75	30.09	27.18	25.01	25.91	25.53
DeepAM$^{(3)}$		**26.35**	**26.19**	**33.46**	**29.47**	**27.73**	**28.89**	**28.68**
19×19 Motion kernel 2								
CSF [13]	GT	29.73	32.12	32.78	27.11	27.19	31.56	30.08
PnP_CNN [31]		31.50	34.89	33.54	31.69	32.89	33.76	33.05
DeepAM$^{(3)}$		**33.05**	**35.72**	**35.86**	**33.21**	**34.14**	**35.12**	**34.52**
CSF [13]	[72]	20.11	26.04	26.92	23.14	21.41	18.87	22.75
PnP_CNN [31]		19.99	27.89	27.56	23.10	21.89	18.06	23.08
DeepAM$^{(3)}$		**22.53**	**30.71**	**30.90**	**24.52**	**23.15**	**20.61**	**25.40**

95×95, respectively. The results from CSF [13] are still blurry (Fig. 7(b)). DAE [32] and PnP_CNN [31] produce deblurred results with ringing artifact (Fig. 7(c-d)). It is because the methods employ the pre-trained CNN denoiser as the proximal mapping instead of training the network in an end-to-end manner. Contrarily, the DeepAM$^{(3)}$ is successfully performed even with inaccurately estimated blur kernels. DeepAM$^{(3)}$ produces promising results regardless of the size of the blur kernels. The kernel size of the second example (95×95) is much larger than the kernel sizes used in the training.

D. Single Image Super-Resolution

For image super-resolution, we consider two image degradation settings, i.e., Gaussian and bicubic downsampling. The former case simulates the low-resolution (LR) images by applying 7×7 Gaussian kernel (as an anti-aliasing filter $\mathbf{B}$) with standard deviation of 1.6 to the high-resolution (HR) images, followed by subsampling. The additive Gaussian noise of $\sigma \in [0, 10]$ is also randomly added to the LR images.

TABLE V

AVERAGE PSNR ON SET5 AND SET14 FOR SINGLE IMAGE SUPER-RESOLUTION WITH GAUSSIAN ($\times 3$) AND BICUBIC DOWNSAMPLING ($\sigma = 0$)

Dataset	σ	Gaussian downsampling ($\times 3$)				
		NCSR [74]	WDnCNN [27]	PnP [17]	PnP_CNN [31]	DeepAM$^{(3)}$
Set5	0	33.02	33.16	32.71	33.45	**33.89**
	2.55	31.76	31.97	31.54	32.98	**33.17**
	7.65	29.74	29.91	29.48	31.01	**31.18**
Set14	0	29.26	29.26	29.18	29.67	**29.72**
	2.55	28.48	28.66	28.47	29.39	**29.48**
	7.65	27.36	27.43	27.06	28.06	**28.23**
Dataset	scale	Bicubic downsampling ($\sigma = 0$)				
		TRD [14]	SRCNN [23]	VDSR [25]	PnP_CNN [31]	DeepAM$^{(3)}$
Set5	$\times 2$	36.86	36.66	37.56	37.43	**37.72**
	$\times 3$	33.18	32.75	33.67	33.39	**33.66**
Set14	$\times 2$	32.51	32.42	33.02	32.88	**33.23**
	$\times 3$	29.43	29.28	29.77	29.61	**29.94**

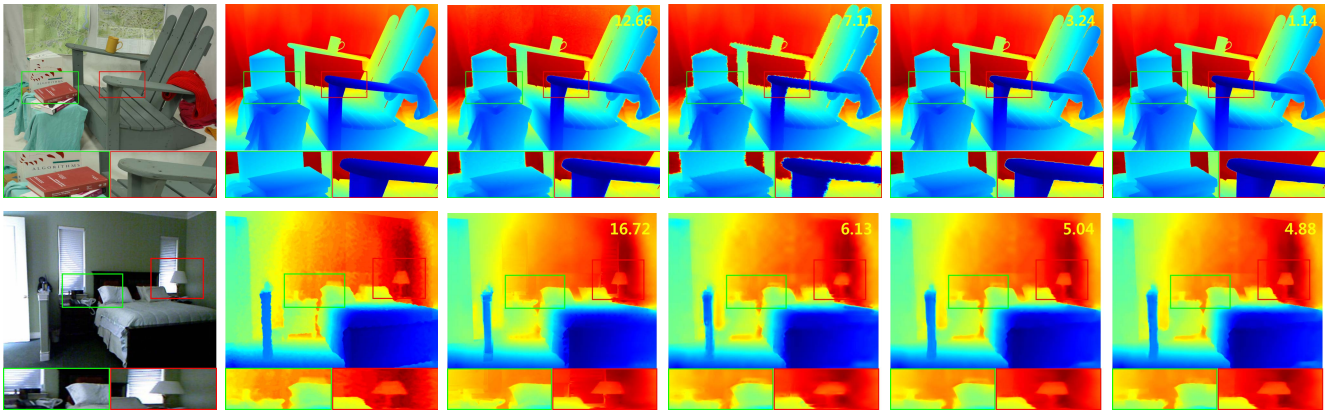
For bicubic downsampling,[6] the reconstruction layer of (8) is similarly implemented with denoising case in (8) as follows:

$$\mathbf{u}^{k+1} = \left(\mathbf{A}^T\mathbf{f} + \gamma^k \mathbf{v}^{k+1}\right) / \left(1 + \gamma^k\right). \tag{23}$$

We trained the DeepAM$^{(3)}$ using 2×10^5 LR/HR patch pairs from 400 training images of BSD dataset [69]. We compare the DeepAM$^{(3)}$ with state-of-the-art SR methods: NCSR [74], SRCNN [23], VDSR [25], WDnCNN [27], PnP [17], and PnP_CNN [31]. The PnP [17] is implemented with the BM3D [19], and the PnP_CNN [31] uses the CNN denoiser as image priors. Since the WDnCNN [27] only considers the bicubic subsampling, we retrained it using the same Gaussian kernel for a fair comparison. Following the literature, a color image is converted into YCbCr color space, and only the luminance channel is super-resolved. The color components are upsampled using the bicubic interpolation.

The test results on Set5 and Set14 [20] are summarized in Table V. It shows that, in terms of PSNR, our DeepAM leads to significant improvements over recent state-of-the-arts methods in all cases. Although the WDnCNN [27] was retrained for the Gaussian kernels, it does not reflect the

[6] In this case, the initial step of (5) takes bicubic upsampled $\mathbf{f}$ ($= \mathbf{A}^T\mathbf{f}$) as the input $\mathbf{u}^0$. To solve the reconstruction layer of (5), we consider the operation of $\mathbf{A}^T\mathbf{A}$ as $\mathbf{I}$.



(a) RGB image (b) ground truth (c) TGV [5] (d) DJF [42] (e) DMSG [43] (f) DeepAM$^{(3)}$

Fig. 9. Depth super-resolution examples (first row : $\times 8$ / second row : $\times 16$): (a) RGB image, (b) ground truth, (c) TGV [5], (d) DJF [42], (e) DMSG [43], and (f) DeepAM$^{(3)}$. The input images are from Middlebury dataset [76] and NYU v2 dataset [75]. BMP ($\delta = 3$) values are included in upper-right.

observation model, leading to limited super-resolution performance. The comparisons with PnP [17] and PnP_CNN [31] clearly demonstrate the effectiveness of our end-to-end training scheme. The regularizer of the PnP scheme [17], [31] is pre-specified (or pre-trained), and thus cannot be jointly optimized with remaining components. Furthermore, the computational complexity is very high as PnP [17] and PnP_CNN [31] require 20 to 30 iterations for convergence, respectively. In contrast, our method achieves superior performance by unrolling the three iterations of AM algorithm [10]. Visual results of Fig. 8 demonstrate that the DeepAM$^{(3)}$ reconstruct much sharper edges and details than other methods (best viewed in electronic version).

TABLE VI

AVERAGE BMP ($\delta = 3$) ON 449 IMAGES FROM THE NYU V2 DATASET [75] AND ON 5 IMAGES FROM THE MIDDLEBURY DATASET [76] FOR DEPTH UPSAMPLING

Method	BMP ($\delta = 3$): NYU v2 [75]/ Middlebury [76]		
	$\times 4$	$\times 8$	$\times 16$
NMRF [6]	1.41 / 3.12	4.21 / 10.71	16.25 / 21.93
TGV [5]	1.58 / 2.74	5.42 / 7.94	17.89 / 18.66
SD filter [7]	1.27 / 2.48	3.56 / 6.69	15.43 / 13.48
DJF [42]	0.68 / 2.39	1.92 / 5.00	5.82 / 10.85
DMSG [43]	0.56 / 1.41	1.55 / 3.45	4.92 / 7.12
DeepAM$^{(3)}$	**0.42 / 1.02**	**1.37 / 2.31**	**4.24 / 5.69**

E. Color-Guided Depth Upsampling

Modern depth sensors, e.g., Microsoft Kinect, provide dense depth measurement in dynamic scene, but typically have a low resolution. A common approach to tackle this problem is to exploit a high-resolution RGB image as guidance. We apply our model to this task, and evaluate it on NYU v2 dataset [75] and Middlebury dataset [76]. The NYU v2 dataset consists of 1449 RGB-D image pairs of indoor scenes, among which 1000 image pairs were used for training and 449 image pairs for testing. Depth values are normalized within $[0, 255]$. To train the network, we randomly collected 2×10^5 RGB-D patch pairs of 64×64 from training set. Following [42], a LR depth image was synthesized by nearest neighbor downsampling (with factors of $\times 4$, $\times 8$, and $\times 16$). We apply DeepAM$^{(3)}$ for color-guided depth super-resolution with the special case of $\mathbf{B} = \mathbf{I}$ in (10).

Fig. 9 shows color-guided depth upsampling results (top: $\times 8$ and bottom: $\times 16$) of TGV [5], DJF [42], DMSG [43] and DeepAM$^{(3)}$. Since DMSG [43] is trained on the small number of RGB/depth images including Middlebury and Sintel MPI dataset [50], we train the model again with our training data for fair comparison. The TGV model [5] uses an anisotropic diffusion tensor that solely depends on the RGB image. The major drawback of this approach is that the RGB-depth coherence assumption is violated in textured surfaces. Thus, the restored depth image is often contaminated by color textures, called texture copying artifacts (Fig. 9(c)). The DJF [42] avoids the texture copying artifacts thanks to faithful CNN responses extracted from both color image and depth map (Fig. 9 (d)). However, this method lacks the regularization constraint that encourages spatial and appearance consistency on the output, and thus it over-smooths the results and does not protect thin structures. Even though the DMSG [43] increases the resolution progressively in the multiple levels leveraging the multi-scale guidance, the upsampled depth maps are not aligned with boundaries of the RGB images well (Fig. 9 (e)). Our DeepAM$^{(3)}$ preserves sharp depth discontinuities without notable artifacts as shown in Fig. 9(f). The quantitative evaluations on the NYU v2 dataset [75] and Middlebury dataset [76][7] are summarized in Table VI. The accuracy was measured by the bad matching percentage (BMP) with tolerance $\delta = 3$.

F. RGB/NIR Restoration

The RGB/NIR restoration aims to enhance a noisy RGB image taken under low-illumination using a spatially aligned NIR image. Applying the proposed method to RGB/NIR restoration poses an additional challenge due to the lack

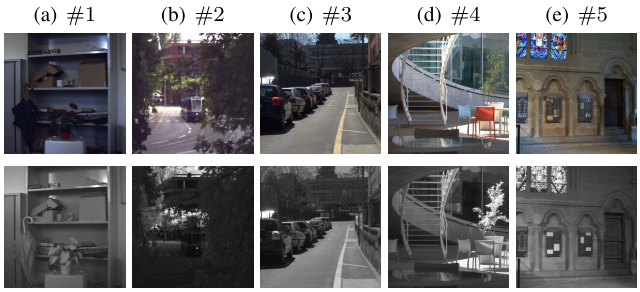
[7]We use 5 examples in the Middlebury dataset [76]: "*Adirondack*", "*Motercycle*", "*Pipes*", "*Playroom*", and "*Flowers*".

(a) NIR image (b) RGB image (c) BM3D [19] (d) Cross-field [8] (e) DJF [42] (f) DeepAM$^{(3)}$

Fig. 10. RGB/NIR restoration for real-world examples: (a) NIR imagse, (b) RGB image, (c) BM3D [19], (d) Cross-field [8], (e) DJF [42], and (f) DeepAM$^{(3)}$. Our DeepAM$^{(3)}$ is trained with blind Gaussian noise in the range of $[10, 55]$.

TABLE VII

THE PSNR RESULTS WITH 5 RGB/NIR PAIRS FROM [78]. THE NOISY RGB IMAGES ARE GENERATED BY ADDING THE SYNTHETIC GAUSSIAN NOISE



(a) #1 (b) #2 (c) #3 (d) #4 (e) #5

Method	PSNR #1	#2	#3	#4	#5	Avg.
Noise Level	$\sigma = 30$					
BM3D [19]	34.11	29.63	30.41	29.36	29.25	30.55
SD filter [7]	31.93	28.39	30.77	28.11	28.41	29.52
Cross-field [8]	32.16	28.79	30.94	28.63	28.74	29.85
DJF [42]	34.28	29.89	31.75	29.16	29.66	30.95
DeepAM$^{(3)}$	**35.94**	**30.10**	**32.04**	**30.60**	**30.17**	**31.77**
Noie Level	$\sigma = 50$					
BM3D [19]	31.74	27.56	28.07	26.90	26.57	28.17
SD filter [7]	30.97	26.13	28.06	25.65	26.11	27.38
Cross-field [8]	31.45	27.59	28.47	26.91	26.98	28.28
DJF [42]	32.11	27.60	29.95	27.10	27.26	28.81
DeepAM$^{(3)}$	**32.71**	**28.13**	**30.29**	**28.34**	**28.14**	**29.49**

TABLE VIII

TEST TIME (IN SECONDS) OF DIFFERENT METHODS IN SINGLE IMAGE DENOISING, DEBLURRING, AND SUPER-RESOLUTION ON 512×512 IMAGES. THE OVERALL RUN TIME IS MEASURED ON GPU

	Single image denoising			
Method	CSF [13]	TRD [14]	DnCNN [26]	DeepAM$^{(3)}$
Time	0.092	0.032	0.051	0.072
	Non-blind deblurring			
Method	CSF [13]	PnP_CNN [31]	DeepAM$^{(3)}$	
Time	0.101	1.123	0.085	
	Single image super-resolution			
Method	WDnCNN [27]	PnP_CNN [31]	DeepAM$^{(3)}$	
Time	0.188	1.668 s	0.102	

of the ground-truth training data. For constructing a large training data, we used the indoor IVRL dataset [77] consisting of 400 RGB/NIR pairs that were recorded under daylight illumination. Specifically, we generated 2×10^5 RGB/NIR patches from 300 image pairs, and added synthetic Gaussian noise with a wide range of $\sigma \in [10, 55]$ to train DeepAM$^{(3)}$.

In Table VII, we performed an objective evaluation using 5 test images in [78]. Since DJF [42] is only trained by RGB/depth for depth refinement, we train the model for RGB/NIR restoration using our training dataset. The DeepAM$^{(3)}$ gives better quantitative results than other state-of-the-art methods. Fig. 10 compares the RGB/NIR restoration results of BM3D [19], cross-field [8], DJF [42], and DeepAM$^{(3)}$ on the real-world examples. All the parameters for [8], [19] were carefully tuned to yield the best visual performance through extensive experiments. The input pairs are taken from the project website of [8]. This experiment demonstrates that the proposed method can be applied to real-world data, although it is trained from the synthetic dataset. It was reported in [78] that the restoration algorithm designed (or trained) to work under a daylight condition could also be used for night condition.

G. Run Time

We compared the test time of DeepAM$^{(3)}$ with recent learning-based restoration methods, which can be implemented through GPU. Table VIII summarizes the runtime of the competing methods in three restoration tasks, i.e., single image image denoising (with $\sigma = 25$), deblurring (with Gaussian blur), and super-resolution (with scale factor of 3). The size of test image is 512×512, and the test time of all methods is measured on GPUs. We use the Nvidia cuDNN-5.1 library to accelerate the CNNs on GPUs. The memory transfer time was not included between CPU and GPU. For overall restoration tasks, the proposed DeepAM$^{(3)}$ achieves very appealing computational efficiency compared to other methods. Especially, DeepAM$^{(3)}$ is much faster than PnP_CNN [31] due to the small number of the AM iterations.

VI. CONCLUSION

In this paper, we have introduced a general framework for image restoration called the DeepAM, which integrates

the CNNs with an energy minimization model. Contrary to the existing data-driven approaches that just produce the restoration result from the CNNs, the DeepAM uses the CNNs to learn the proximal mapping in the AM algorithm. Since we do not presume the point-wise separable regularization functions, our aggregated mapping is more flexible to describe complex structures of natural images. We have derived efficient forward and backward propagation for the reconstruction layers depending on the observation models, which enables optimizing the overall parameters in an end-to-end manner. With unrolling only three times of AM iterations, the DeepAM outperforms nonlocal-based methods, Plug-and-Play regularization, and the CNN-based approaches in various image restoration applications.

APPENDIX

A. $\frac{\partial \mathcal{L}}{\partial \gamma}$ for Deblurring

Differentiating both sides of (7) with respect to γ, we can obtain the following expression:

$$\mathbf{u}^T \left(\frac{\partial \mathbf{L}}{\partial \gamma} \right)^T + \frac{\partial \mathbf{u}}{\partial \gamma} \mathbf{L}^T = \mathbf{v}^T. \tag{24}$$

Since the off-diagonal components of $\mathbf{L}$ do not depend on γ, it is straightforward to obtain $\frac{\partial \mathbf{u}}{\partial \gamma}$:

$$\frac{\partial \mathbf{u}}{\partial \gamma} = (\mathbf{v} - \mathbf{u})^T \mathbf{L}^{-1}. \tag{25}$$

Substituting (25) into (15), we have:

$$\frac{\partial \mathcal{L}}{\partial \gamma} = (\mathbf{v} - \mathbf{u})^T \left(\mathbf{L}^{-1} \frac{\partial \mathcal{L}}{\partial \mathbf{u}} \right). \tag{26}$$

Thus, the backward steps for γ can be efficiently performed using the FFT.

B. $\frac{\partial \mathcal{L}}{\partial \gamma}$ for Super-Resolution

Denoting $\mathbf{P} = (\gamma \mathbf{I} + \mathbf{A}\mathbf{A}^T)^{-1}$ and $\mathbf{K} = \mathbf{A}^T \mathbf{P} \mathbf{A}$. $\frac{\partial \mathbf{K}}{\partial \gamma}$ in (21) is:

$$\frac{\partial \mathbf{K}}{\partial \gamma} = \mathbf{A}^T \frac{\partial \mathbf{P}}{\partial \gamma} \mathbf{A}. \tag{27}$$

With the identity of $\frac{\partial \mathbf{A}^{-1}}{\partial \gamma} = -\mathbf{A}^{-1} \frac{\partial \mathbf{A}}{\partial \gamma} \mathbf{A}^{-1}$, $\frac{\partial \mathbf{P}}{\partial \gamma}$ is then obtained as follows:

$$\begin{aligned} \frac{\partial \mathbf{P}}{\partial \gamma} &= -\mathbf{P} \frac{\partial (\gamma \mathbf{I} + \mathbf{A}\mathbf{A}^T)}{\partial \gamma} \mathbf{P} \\ &= -\mathbf{P}^2 \end{aligned} \tag{28}$$

Substituting (27) and (28) into (21) yields:

$$\frac{\partial \mathbf{u}}{\partial \gamma} = \frac{1}{\gamma} \left\{ \mathbf{z}^T \mathbf{A}^T \mathbf{P}^2 \mathbf{A} + \mathbf{v}^T (\mathbf{I} - \mathbf{K}) - \mathbf{u}^T \right\}, \tag{29}$$

from which we finally conclude:

$$\frac{\partial \mathcal{L}}{\partial \gamma} = \frac{1}{\gamma} \left\{ \mathbf{z}^T \mathbf{A}^T \mathbf{P}^2 \mathbf{A} + \mathbf{v}^T (\mathbf{I} - \mathbf{K}) - \mathbf{u}^T \right\} \frac{\partial \mathcal{L}}{\partial \mathbf{u}}. \tag{30}$$

Again, we can efficiently implement (30) in closed-form using the polyphase decomposition.

REFERENCES

[1] L. I. Rudin, S. Osher, and E. Fatemi, "Nonlinear total variation based noise removal algorithms," *Phys. D: Nonlinear Phenomena*, vol. 60, nos. 1–4, pp. 259–268, Nov. 1992.

[2] D. Krishnan and R. Fergus, "Fast image deconvolution using hyper-Laplacian priors," in *Proc. Adv. Neural Inf. Process. Syst.*, 2009, pp. 1033–1041.

[3] M. Irani and S. Peleg, "Motion analysis for image enhancement: Resolution, occlusion, and transparency," *J. Vis. Commun. Image Represent.*, vol. 4, no. 4, pp. 324–335, Dec. 1993.

[4] A. N. Tikhonov, "Solution of incorrectly formulated problems and the regularization method," *Sov. Math.*, vol. 4, pp. 1035–1038, 1963.

[5] D. Ferstl, C. Reinbacher, R. Ranftl, M. Ruether, and H. Bischof, "Image guided depth upsampling using anisotropic total generalized variation," in *Proc. IEEE Int. Conf. Comput. Vis.*, Dec. 2013, pp. 993–1000.

[6] J. Park, H. Kim, Y.-W. Tai, M. S. Brown, and I. Kweon, "High quality depth map upsampling for 3D-TOF cameras," in *Proc. Int. Conf. Comput. Vis.*, Nov. 2011, pp. 1623–1630.

[7] B. Ham, M. Cho, and J. Ponce, "Robust guided image filtering using nonconvex potentials," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 40, no. 1, pp. 192–207, Jan. 2018.

[8] X. Shen, Q. Yan, L. Xu, L. Ma, and J. Jia, "Multispectral joint image restoration via optimizing a scale map," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 37, no. 12, pp. 2518–2530, Dec. 2015.

[9] C. Feng, S. Zhuo, X. Zhang, L. Shen, and S. Susstrunk, "Near-infrared guided color image dehazing," in *Proc. IEEE Int. Conf. Image Process.*, Sep. 2013, pp. 2363–2367.

[10] Y. Wang, J. Yang, W. Yin, and Y. Zhang, "A new alternating minimization algorithm for total variation image reconstruction," *SIAM J. Imag. Sci.*, vol. 1, no. 3, pp. 248–272, Jan. 2008.

[11] S. Yun and H. Woo, "Linearized proximal alternating minimization algorithm for motion deblurring by nonlocal regularization," *Pattern Recognit.*, vol. 44, no. 6, pp. 1312–1326, Jun. 2011.

[12] N. Parikh and S. Boyd, "Proximal algorithms," *Found. Trends Optim.*, vol. 1, no. 3, pp. 123–231, 2013.

[13] U. Schmidt and S. Roth, "Shrinkage fields for effective image restoration," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2014, pp. 2774–2781.

[14] Y. Chen and T. Pock, "Trainable nonlinear reaction diffusion: A flexible framework for fast and effective image restoration," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 6, pp. 1256–1272, Jun. 2017.

[15] S. V. Venkatakrishnan, C. A. Bouman, and B. Wohlberg, "Plug-and-Play priors for model based reconstruction," in *Proc. IEEE Global Conf. Signal Inf. Process.*, Dec. 2013, pp. 945–948.

[16] A. Brifman, Y. Romano, and M. Elad, "Turning a denoiser into a super-resolver using plug and play priors," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Sep. 2016, pp. 1404–1408.

[17] S. H. Chan, X. Wang, and O. A. Elgendy, "Plug-and-Play ADMM for image restoration: Fixed-point convergence and applications," *IEEE Trans. Comput. Imag.*, vol. 3, no. 1, pp. 84–98, Mar. 2017.

[18] A. Buades, B. Coll, and J.-M. Morel, "A non-local algorithm for image denoising," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2005, pp. 60–65.

[19] K. Dabov, A. Foi, V. Katkovnik, and K. Egiazarian, "Image denoising by sparse 3-D transform-domain collaborative filtering," *IEEE Trans. Image Process.*, vol. 16, no. 8, pp. 2080–2095, Aug. 2007.

[20] R. Timofte, V. De Smet, and L. V. Gool, "A+: Adjusted anchored neighborhood regression for fast super-resolution," in *Proc. Asian Conf. Comput. Vis.*, 2014, pp. 111–126.

[21] H. C. Burger, C. J. Schuler, and S. Harmeling, "Image denoising: Can plain neural networks compete with BM3D?" in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2012, pp. 2392–2399.

[22] J. Xie, L. Xu, and E. Chen, "Image denoising and inpainting with deep neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2012, pp. 341–349.

[23] C. Dong, C. C. Loy, K. He, and X. Tang, "Learning a deep convolutional network for image super-resolution," in *Proc. Eur. Conf. Comput. Vis.*, 2014, pp. 184–199.

[24] L. Xu, J. S. Ren, C. Liu, and J. Jia, "Deep convolutional neural network for image deconvolution," in *Proc. Adv. Neural Inf. Process. Syst.*, 2014, pp. 1790–1798.

[25] J. Kim, J. K. Lee, and K. M. Lee, "Accurate image super-resolution using very deep convolutional networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 1646–1654.

[26] 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.

[27] W. Bae, J. Yoo, and J. C. Ye, "Beyond deep residual learning for image restoration: Persistent homology-guided manifold simplification," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW)*, Jul. 2017, pp. 145–153.

[28] R. Ranftl and T. Pock, "A deep variational model for image segmentation," in *Proc. German Conf. Pattern Recognit.*, 2014, pp. 107–118.

[29] G. Riegler, M. Rüther, and H. Bischof, "ATGV-net: Accurate depth super-resolution," in *Proc. Eur. Conf. Comput. Vis.*, 2016, pp. 268–284.

[30] G. Riegler, D. Ferstl, M. Rüther, and H. Bischof, "A deep primal-dual network for guided depth super-resolution," in *Proc. Brit. Mach. Vis. Conf.*, 2016, p. 1.

[31] K. Zhang, W. Zuo, S. Gu, and L. Zhang, "Learning deep CNN denoiser prior for image restoration," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 3929–3938.

[32] S. Arjomand Bigdeli and M. Zwicker, "Image restoration using autoencoding priors," 2017, *arXiv:1703.09964*. [Online]. Available: http://arxiv.org/abs/1703.09964

[33] T. Meinhardt, M. Moeller, C. Hazirbas, and D. Cremers, "Learning proximal operators: Using denoising networks for regularizing inverse imaging problems," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 1781–1790.

[34] Y. Kim, H. Jung, D. Min, and K. Sohn, "Deeply aggregated alternating minimization for image restoration," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jul. 2017, pp. 6419–6427.

[35] M. J. Black, G. Sapiro, D. H. Marimont, and D. Heeger, "Robust anisotropic diffusion," *IEEE Trans. Image Process.*, vol. 7, no. 3, pp. 421–432, Mar. 1998.

[36] K. Bredies, K. Kunisch, and T. Pock, "Total generalized variation," *SIAM J. Imag. Sci.*, vol. 3, no. 3, pp. 492–526, 2010.

[37] L. Xu, C. Lu, Y. Xu, and J. Jia, "Image smoothing via L_0 gradient minimization," *ACM Trans. Graph.*, vol. 30, no. 6, p. 174, 2011.

[38] W. Dong, G. Shi, Y. Ma, and X. Li, "Image restoration via simultaneous sparse coding: Where structured sparsity meets Gaussian scale mixture," *Int. J. Comput. Vis.*, vol. 144, nos. 2–3, pp. 217–232, 2015.

[39] Y. Li, W. Dong, X. Xie, G. Shi, J. Wu, and X. Li, "Image super-resolution with parametric sparse model learning," *IEEE Trans. Image Process.*, vol. 27, no. 9, pp. 4638–4650, Sep. 2018.

[40] A. Agrawal, R. Raskar, S. K. Nayar, and Y. Li, "Removing photography artifacts using gradient projection and flash-exposure sampling," *ACM Trans. Graph.*, vol. 24, no. 3, pp. 828–835, Jul. 2005.

[41] 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.

[42] Y. Li, J. Huang, N. Ahuja, and M. Yang, "Deep joint image filtering," in *Proc. Eur. Conf. Comput. Vis.*, 2016, pp. 154–169.

[43] T. Hui, C. Loy, and X. Tang, "Depth map super-resolution by deep multi-scale guidance," in *Proc. Eur. Conf. Comput. Vis.*, 2016, pp. 353–369.

[44] S. Zheng *et al.*, "Conditional random fields as recurrent neural networks," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Dec. 2015, pp. 1529–1537.

[45] Y. Yang, J. Sun, H. Li, and Z. Xu, "ADMM-CSNet: A deep learning approach for image compressive sensing," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 42, no. 3, pp. 521–538, Mar. 2020.

[46] Z. Milacski, B. Poczos, and A. Lorincz, "Differentiable unrolled alternating direction method of multipliers for onenet," in *Proc. Brit. Mach. Vis. Conf.*, 2018, p. 140.

[47] S. Diamond, V. Sitzmann, F. Heide, and G. Wetzstein, "Unrolled optimization with deep priors," 2017, *arXiv:1705.08041*. [Online]. Available: http://arxiv.org/abs/1705.08041

[48] W. Dong, P. Wang, W. Yin, G. Shi, F. Wu, and X. Lu, "Denoising prior driven deep neural network for image restoration," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 41, no. 10, pp. 2305–2318, Oct. 2019.

[49] J. Chang, C. Li, B. Poczos, B. Kumar, and A. Sankaranarayanan, "One network to solve them all–solving linear inverse problems using deep projection models," in *Proc. IEEE Int. Conf. Comput. Vis.*, Oct. 2017, pp. 5888–5897.

[50] D. Butler, J. Wulff, G. Stanly, and M. Black, "A naturalistic open source movie for optical flow evaluation," in *Proc. Eur. Conf. Comput. Vis.*, 2012, pp. 611–625.

[51] K. T. Fang and Y. T. Zhang, *Generalized Multivariate Analysis*. Beijing, China: Science Press, 1990.

[52] A. Beck and M. Teboulle, "Gradient-based algorithms with applications to signal recovery problems," in *Convex Optimization in Signal Processing and Communications*. Cambridge, U.K.: Cambridge Univ. Press, 2009, pp. 42–88.

[53] Y. Nesterov, "A method of solving a convex programming problem with convergence rate $O(1/k^2)$," *Sov. Math. Dokl*, vol. 27, no. 3, pp. 372–376, 1983.

[54] M. V. Afonso, J. M. Bioucas-Dias, and M. A. T. Figueiredo, "Fast image recovery using variable splitting and constrained optimization," *IEEE Trans. Image Process.*, vol. 19, no. 9, pp. 2345–2356, Sep. 2010.

[55] M. Nikolova and M. K. Ng, "Analysis of half-quadratic minimization methods for signal and image recovery," *SIAM J. Sci. Comput.*, vol. 27, no. 3, pp. 937–966, Jan. 2005.

[56] J. M. Bioucas-Dias and M. A. T. Figueiredo, "An iterative algorithm for linear inverse problems with compound regularizers," in *Proc. 15th IEEE Int. Conf. Image Process.*, Oct. 2008, pp. 685–688.

[57] Y. Kim, B. Ham, C. Oh, and K. Sohn, "Structure selective depth superresolution for RGB-D cameras," *IEEE Trans. Image Process.*, vol. 25, no. 11, pp. 5227–5238, Nov. 2016.

[58] W. J. Duncan, "Some devices for the solution of large sets of simultaneous equations," *Phil. Mag. J. Sci.*, vol. 35, pp. 660–670, 1944.

[59] J. Yang, W. Yin, Y. Zhang, and Y. Wang, "A fast algorithm for edge-preserving variational multichannel image restoration," *SIAM J. Imag. Sci.*, vol. 2, no. 2, pp. 569–592, Jan. 2009.

[60] J. Liu, S. Zhang, S. Wang, and D. Metaxas, "Multispectral deep neural networks for pedestrian detection," in *Proc. Brit. Mach. Vis. Conf.*, 2016, pp. 1–13.

[61] S. Gupta, R. Girshick, P. Arbelaez, and J. Malik, "Learning rich features from RGB-D images for object detection and segmentation," in *Proc. Eur. Conf. Comput. Vis.*, 2014, pp. 345–360.

[62] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," 2015, *arXiv:1502.03167*. [Online]. Available: https://arxiv.org/abs/1502.03167

[63] S. Gu, L. Zhang, W. Zuo, and X. Feng, "Weighted nuclear norm minimization with application to image denoising," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2014, pp. 2862–2869.

[64] D. Zoran and Y. Weiss, "From learning models of natural image patches to whole image restoration," in *Proc. Int. Conf. Comput. Vis.*, Nov. 2011, pp. 479–486.

[65] 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.

[66] G. Schaefer and M. Stich, "UCID: An uncompressed color image database," *Int. Soc. Opt. Photon., Electron. Imag.*, pp. 472–480, 2003.

[67] 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.

[68] 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.

[69] D. Martin, C. Fowlkes, D. Tal, and J. Malik, "A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics," in *Proc. 8th IEEE Int. Conf. Comput. Vis. (ICCV)*, Jul. 2001, pp. 416–423.

[70] H. Zhang, V. Sindagi, and V. M. Patel, "Image de-raining using a conditional generative adversarial network," *IEEE Trans. Circuits Syst. Video Technol.*, 2019.

[71] A. Levin, Y. Weiss, F. Durand, and W. T. Freeman, "Understanding and evaluating blind deconvolution algorithms," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2009, pp. 1964–1971.

[72] S. Cho and S. Lee, "Fast motion deblurring," *ACM Trans. Graph.*, vol. 28, no. 5, p. 145, 2009.

[73] L. Xu and J. Jia, "Two-phase kernel estimation for robust motion deblurring," in *Proc. Eur. Conf. Comput. Vis.*, 2010, pp. 157–170.

[74] W. Dong, L. Zhang, G. Shi, and X. Li, "Nonlocally centralized sparse representation for image restoration," *IEEE Trans. Image Process.*, vol. 22, no. 4, pp. 1620–1630, Apr. 2013.

[75] N. Silberman, D. Hoiem, P. Kohli, and R. Fergus, "Indoor segmentation and support inference from RGBD images," in *Proc. Eur. Conf. Comput. Vis.*, 2012, pp. 746–760.

[76] D. Scharstein *et al.*, "High-resolution stereo datasets with subpixel-accurate ground truth," in *Proc. German Conf. Pattern Recognit.*, 2014, pp. 31–42.

[77] N. Salamanti, D. Larlus, G. Csurka, and S. Susstrunk, "Incorporating near-infrared information into semantic image segmentation," 2014, *arXiv:1406.6147*. [Online]. Available: https://arxiv.org/abs/1406.6147

[78] H. Honda, R. Timofte, and L. Van Gool, "Make my day-high-fidelity color denoising with near-infrared," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR) Workshops*, Jun. 2015, pp. 82–90.

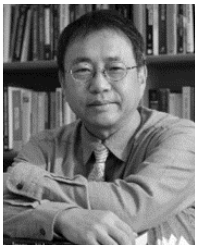
Hyungjoo Jung (Member, IEEE) received the B.S. degree in electrical and electronic engineering from Yonsei University, Seoul, South Korea, in 2016, where he is currently pursuing the joint M.S. and Ph.D. degrees in electrical and electronic engineering. His current research interests include 2D/3D video processing, computational photography, and machine learning.

Youngjung Kim (Member, IEEE) received the B.S. and Ph.D. degrees in electrical and electronic engineering from Yonsei University in 2013 and 2018, respectively. He is currently a Senior Researcher with the Institute of Defense Advanced Technology Research, Agency for Defense Development, Daejeon, South Korea. His current research interests include variational methods, continuous optimization, and machine learning, both in theory and applications in image processing and computer vision.

Dongbo Min (Senior Member, IEEE) received the B.S., M.S., and Ph.D. degrees from the School of Electrical and Electronic Engineering, Yonsei University, Seoul, South Korea, in 2003, 2005, and 2009, respectively. From 2009 to 2010, he was a Postdoctoral Researcher with Mitsubishi Electric Research Laboratories, Cambridge, MA, USA. From 2010 to 2015, he was with Advanced Digital Sciences Center, Singapore. From 2015 to 2018, he was an Assistant Professor with the Department of Computer Science and Engineering, Chungnam National University, Daejeon, South Korea. Since 2018, he has been with the Department of Computer Science and Engineering, Ewha Womans University, Seoul. His current research interests include computer vision and deep learning.

Hyunsung Jang (Member, IEEE) received the B.S. and M.S. degrees in electrical and computer engineering from Pusan National University, South Korea, in 2007 and 2009, respectively. He is currently pursuing the Ph.D. degree in electrical and electronic engineering with Yonsei University, Seoul, South Korea. He is also a Chief Research Engineer with the EO/IR R&D Laboratory, LIG Nex1 Company Ltd. His research interests include computer vision, image processing, pattern recognition, and machine learning.

Namkoo Ha (Member, IEEE) received the Ph.D. degree in computer engineering from Kyungpook National University, South Korea, in 2008. He is currently a Chief Research Engineer with the EO/IR R&D Laboratory, LIG Nex1 Company Ltd. His current research interest includes developing intelligent EO/IR systems through the use of deep learning.



Kwanghoon Sohn (Senior Member, IEEE) received the B.E. degree in electronic engineering from Yonsei University, Seoul, South Korea, in 1983, the M.S.E.E. degree in electrical engineering from the University of Minnesota, Minneapolis, MN, USA, in 1985, and the Ph.D. degree in electrical and computer engineering from North Carolina State University, Raleigh, NC, USA, in 1992. He was a Senior Member of Research Engineering with Satellite Communication Division, Electronics and Telecommunications Research Institute, Daejeon, South Korea, from 1992 to 1993, and a Postdoctoral Fellow with the MRI Center, Medical School of Georgetown University, Washington, DC, USA, in 1994. He was a Visiting Professor with Nanyang Technological University, Singapore, from 2002 to 2003. He is currently an Underwood Distinguished Professor with the School of Electrical and Electronic Engineering, Yonsei University. His research interests include 3D image processing and computer vision.