Deep Graph Laplacian Regularization for Robust Denoising of Real Images
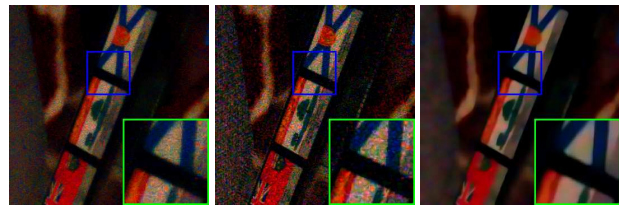
Jin Zeng[1*]   Jiahao Pang[1*]   Wenxiu Sun[1]   Gene Cheung[2]

[1]SenseTime Research   [2]Department of EECS, York University

{zengjin, pangjiahao, sunwenxiu}@sensetime.com, genec@yorku.ca

Abstract

Recent developments in deep learning have revolutionized the paradigm of image restoration. However, its applications on real image denoising are still limited, due to its sensitivity to training data and the complex nature of real image noise. In this work, we combine the robustness merit of model-based approaches and the learning power of data-driven approaches for real image denoising. Specifically, by integrating graph Laplacian regularization as a trainable module into a deep learning framework, we are less susceptible to overfitting than pure CNN-based approaches, achieving higher robustness to small datasets and cross-domain denoising. First, a sparse neighborhood graph is built from the output of a convolutional neural network (CNN). Then the image is restored by solving an unconstrained quadratic programming problem, using a corresponding graph Laplacian regularizer as a prior term. The proposed restoration pipeline is fully differentiable and hence can be end-to-end trained. Experimental results demonstrate that our work is less prone to overfitting given small training data. It is also endowed with strong cross-domain generalization power, outperforming the state-of-the-art approaches by a remarkable margin.

1. Introduction

Image denoising is the most fundamental image restoration problem, which has been studied for decades. In order to regularize its ill-posed nature, a large body of works adopt *signal priors*. By adopting a certain image model, one assumes that the original image should induce a small value for a given *model-based* signal prior. Representative priors in the literature include non-local self-similarity [5], total variation (TV) prior [37], sparsity prior [15], graph Laplacian regularizer [33], *etc*. However, these works place their emphases to the removal of additive white Gaussian noise (AWGN), which is unrealistic and limits their applications in practice. In the real world, image noise stems from multiple sources, *e.g.*, thermal noise, shot noise, dark

---

*Both authors contributed equally to this work. Jiahao Pang is the corresponding author.

(a) Noise Clinic.   (b) CDnCNN.   (c) DeepGLR.

Figure 1. Results of real image denoising. (a) Noise Clinic (model-based); (b) CDnCNN (data-driven); (c) DeepGLR (proposed). Our method and CDnCNN are trained for Gaussian denoising.

current noise, making it much more sophisticated than the ideal AWGN.

Recent developments in *deep learning* have revolutionized the aforementioned model-based paradigm in image restoration. Thanks to the strong learning capacity of convolutional neural networks (CNN) to capture image characteristics, CNN-based approaches have achieved the state-of-the-art performance in Gaussian denoising, *e.g.*, [48, 43, 40]. However, the application of deep learning models on real noise removal remains quite challenging. Unlike model-based approaches, CNN-based approaches are data-driven. To learn a CNN for real image noise removal, thousands of real noisy images and their noise-free versions are necessary to characterize the correspondence between the corrupted images and the ground-truths [50]. Unfortunately, acquiring the noise-free images is non-trivial [46, 7], leading to limited amount of training data. In this case, a purely data-driven approach is prone to overfit to the particular characteristics of the training data. It fails on test images with statistics different from the training images [29], *e.g.*, Figure 1b showcases the result of a pure data-driven approach trained for a different domain.

Differently, model-based denoising approaches rely on basic assumptions about the original images, which "encode" assumed image characteristics. Without the notion of training, the performance of model-based denoising is generally more robust than data-driven approaches when facing the heterogeneity of natural images [13]. However, the assumed characteristics may not perfectly hold in the real

world, hindering their performance and flexibility in practice [30], *e.g.*, the denoising result of Figure 1a.

To achieve robust denoising of real images, in this paper we combine the robustness merit of model-based approaches and the powerful learning capacity of data-driven approaches. We achieve this goal by incorporating the graph Laplacian regularizer—a simple yet effective image prior for image restoration tasks—into a deep learning framework. Specifically, we train a CNN which takes as input a real noisy image and outputs a set of feature maps. Subsequently, a neighborhood graph is built from the output features. The image is then denoised by solving an unconstrained quadratic programming (QP) problem, assuming that the underlying true image induces a small value of graph Laplacian regularizer. Figure 1c shows the denoising result of our approach, one may clearly see its superiority to the competing methods.

The contributions of our work are as follows:

 (i) We are the first in literature to incorporate the widely used graph Laplacian regularizer into deep neural networks as a *fully-differentiable* layer, extracting underlying features of the input noisy images and boosting the performance of the subsequent restoration.

 (ii) Our architecture couples the strong graph Laplacian regularization layer—an adaptive low-pass linear filter regardless of the training data—with a light-weight CNN for pre-filtering, making our approach less susceptible to overfitting. Moreover, by constraining the regularization weight to prevent steep local minimum, our pipeline is provably *numerical stable*.

(iii) Experimentation shows that, our approach achieves *robust* real noise removal in terms of two perspectives. Given small amount of training data, our proposal outperforms CNN-based approaches by avoiding overfitting. Secondly, it exhibits strong cross-domain generalization ability, *e.g.*, our framework training for Gaussian denoising performs reasonably well on real image denoising.

We call our proposal *deep graph Laplacian regularization*, or DeepGLR for short. This paper is organized as follows. Related works are reviewed in Section 2. We then present our DeepGLR denoising framework combining CNN and a differentiable graph Laplacian regularization layer in Section 3. Section 4 presents the experimentation and Section 5 concludes our work.

2. Related Works

We first review several deep learning models for image restoration while focusing on image denoising. We then turn to the review of several representative signal priors,

with a focus on graph Laplacian regularization. We also briefly review a few works on graph learning.

CNN-based image denoising: CNN-based approaches were first popularized in high-level vision tasks, *e.g.*, classification [24] and detection [32], then gradually penetrated into low-level restoration tasks such as image denoising [48], super-resolution [12], and non-blind deblurring [47]. To address the problem of Gaussian noise removal with CNN, Zhang *et al.* [48] utilize residual learning and batch normalization to build a deep architecture, which provides state-of-the-art results. In [21], Jain *et al.* propose a simple network for natural image denoising and relate it to Markov random field (MRF) methods. To build a CNN capable of handling several noise levels, Vemulapalli *et al.* [43] employ conditional random field (CRF) for regularization. Other related works on denoising with CNN includes [28, 40, 49, 26, 8], *etc*. Despite their good performance, these approaches focuses on Gaussian denoising and have strong dependency on the training data. For effective real image denoising and enhancing, Chen *et al.* [7] train a CNN to directly perform restoration on raw image data. Differently, our DeepGLR enhances the robustness of the denoising pipeline, so as to achieve effective real noise removal.

Image denoising with signal priors: We hereby review a few representative works on image denoising using signal priors. For a more complete review, we refer the readers to [30]. In [5], Buades *et al.* assume that similar image patches recur non-locally throughout an image. Such a self-similarity assumption has been adopted in many subsequent proposals. One notable method, block-matching 3-D (BM3D) [10], performs 3-D transform and Wiener filtering on the grouped similar patches. Elad *et al.* [15] propose K-SVD denoising, which seeks sparse representations to describe noiseless patches with a learned dictionary. A very recent work [45] extents this notion for real image denoising, though its complexity is too high for practical usage.

Graph Laplacian regularization is a recent popular image prior in the literature, *e.g.*, [33, 16, 17]. Despite its simplicity, graph Laplacian regularization performs reasonably well for many restoration tasks [30]. It assumes that the original image, denoted as $\mathbf{x} \in \mathbb{R}^m$, is smooth with respect to an appropriately chosen graph $\mathcal{G}$. Specifically, it imposes that the value of $\mathbf{x}^\mathrm{T}\mathbf{L}\mathbf{x}$, *i.e.*, the graph Laplacian regularizer, should be small for the original image $\mathbf{x}$, where $\mathbf{L} \in \mathbb{R}^{m \times m}$ is the Laplacian matrix of graph $\mathcal{G}$. Typically, a graph Laplacian regularizer is employed for a quadratic programming (QP) formulation [33, 20, 27]. Nevertheless, choosing a proper graph for image restoration remains an open question. In [16, 27], the authors build their graphs from the corrupted image with simple ad-hoc rules; while in [33], Pang *et al.* derive sophisticated principles for building graphs under strict conditions. Different from existing works, our DeepGLR framework constructs neighbor-

hood graphs from the CNN outputs, *i.e.,* our graphs are built in a *data-driven* manner, which learns the appropriate graph connectivity for restoration directly. In [38, 4], the authors also formulate graph Laplacian regularization in a deep learning pipeline; yet unlike ours, their graph constructions are fixed functions, *i.e.*, they are not data-driven.

Learning with graphs: there exist a few works combining tools of graph theory with data-driven approaches. In [23, 11] and subsequent works, the authors study the notion of convolution on graphs, which enables CNNs to be applied on irregular graph kernels. In [41], Turaga *et al.* let a CNN to directly output edge weights for fixed graphs; while Egilmez *et al.* [14] learn the graph Laplacian matrices with a maximum a posteriori (MAP) formulation. Our work also learns the graph structure. Different from the methodology of existing works, we build the graphs from the learned features of CNN for subsequent regularizations.

3. Deep Graph Laplacian Regularization

We now present our DeepGLR framework integrating graph Laplacian regularization into CNN for real noise removal. A graph Laplacian regularization layer is composed of two modules: a graph construction module [33] and a QP solver [2]. We first present the details of graph Laplacian regularization [33, 16, 20] as an image prior, then introduce its encapsulation as a layer in a CNN.

3.1. Formulation

We start our illustration with a simple AWGN denoising formulation, which will be extended to take account for more complex cases (Section 3.3). Consider the following image corruption model:

$$\mathbf{y} = \mathbf{x} + \mathbf{n}, \tag{1}$$

Here $\mathbf{x} \in \mathbb{R}^m$ is the original image or image patch (in vector form) with m pixels, while $\mathbf{n}$ is an additive Gaussian noise term and $\mathbf{y}$ is the noisy observation. Given an appropriate neighborhood graph $\mathcal{G}$ with m vertices representing the pixels, graph Laplacian regularization assumes the original image $\mathbf{x}$ is *smooth* with respect to $\mathcal{G}$ [39]. Denoting the edge weight connecting pixels i and j as w_{ij}, the adjacency matrix $\mathbf{A}$ of graph $\mathcal{G}$ is an m-by-m matrix, whose (i,j)-th entry is w_{ij}. The degree matrix of $\mathcal{G}$ is a diagonal matrix $\mathbf{D}$ whose i-th diagonal entry is $\sum_{j=1}^{m} w_{ij}$. Then the (combinatorial) graph Laplacian matrix $\mathbf{L}$ is a positive semidefinite (PSD) matrix given by $\mathbf{L} = \mathbf{D} - \mathbf{A}$, which induces the graph Laplacian regularizer $\mathbf{x}^\mathrm{T}\mathbf{L}\mathbf{x} \geq 0$ [39].

To recover $\mathbf{x} \in \mathbb{R}^m$ with graph Laplacian regularization, one can formulate a *maximum a posteriori* (MAP) problem as follows:

$$\mathbf{x}^\star = \underset{\mathbf{x}}{\arg\min} \|\mathbf{y} - \mathbf{x}\|_2^2 + \mu \cdot \mathbf{x}^\mathrm{T}\mathbf{L}\mathbf{x}, \tag{2}$$

where the first term is a fidelity term (negative log likelihood) computing the difference between the observation $\mathbf{y}$ and the recovered signal $\mathbf{x}$, and the second term is the graph Laplacian regularizer (negative log signal prior). $\mu \geq 0$ is a weighting parameter. For effective regularization, one needs an appropriate graph $\mathcal{G}$ reflecting the image structure of ground-truth $\mathbf{x}$. In most works such as [33, 20, 31], it is derived from the noisy $\mathbf{y}$ or a pre-filtered version of $\mathbf{y}$.

For illustration, we define a matrix-valued function $\mathbf{F}(\mathbf{y}) : \mathbb{R}^m \mapsto \mathbb{R}^{m \times N}$, where its n-th column is denoted as $\mathbf{f}_n$ where $\mathbf{f}_n \in \mathbb{R}^m$, $1 \leq n \leq N$. Hence, applying $\mathbf{F}$ to observation $\mathbf{y}$ maps it to a set of N length-m vectors $\{\mathbf{f}_n\}_{n=1}^N$. Using the same terminology in [33], the $\mathbf{f}_n$'s are called *exemplars*. Then the edge weight w_{ij} ($1 \leq i, j \leq m$) is computed by:

$$w_{ij} = \exp\left(-\frac{\mathrm{dist}(i,j)}{2\epsilon^2}\right), \tag{3}$$

where

$$\mathrm{dist}(i,j) = \sum_{n=1}^{N} \left(\mathbf{f}_n(i) - \mathbf{f}_n(j)\right)^2. \tag{4}$$

Here $\mathbf{f}_n(i)$ denotes the i-th element of $\mathbf{f}_n$. (4) is the Euclidean distance between pixels i and j in the N-dimension feature space defined by $\{\mathbf{f}_n\}_{n=1}^N$. In practice, the $\mathbf{f}_n$'s should reflect the characteristics of the ground-truth image $\mathbf{x}$ for effective restoration. Though different works use different schemes to build a similarity graph $\mathcal{G}$, most of them differ only in the choice of exemplars $\mathbf{F}(\mathbf{y})$ (or the $\mathbf{f}_n$'s). In [27, 22], the authors restrict the graph structure to be a 4-connected grid graph and let $\mathrm{dist}(i,j) = (\mathbf{y}(i) - \mathbf{y}(j))^2$, which is equivalent to let $\mathbf{F}(\mathbf{y}) = \mathbf{y}$. In [20], Hu *et al.* operate on overlapping patches and let $\mathbf{F}(\mathbf{y})$ be the noisy patches similar to $\mathbf{y}$. Pang *et al.* [33] interpret the $\{\mathbf{f}_n\}_{n=1}^N$ as samples on a high-dimensional Riemannian manifold and derive the optimal $\mathbf{F}$ under certain assumptions.

3.2. Graph Laplacian Regularization Layer

In contrast to existing works, we deploy graph Laplacian regularization as a layer in a deep learning pipeline, by *implementing the function $\mathbf{F}$ with a CNN*. In other words, the corrupted observation $\mathbf{y}$ is fed to a CNN (denoted as $\mathrm{CNN}_\mathbf{F}$) which outputs N exemplars (or feature maps) $\{\mathbf{f}_n\}_{n=1}^N$.

Specifically, we perform denoising on a *patch-by-patch* basis, similarly done in [33, 20, 27]. Suppose the observed noisy image, denoted as $\mathcal{Y}$, is divided into K overlapping patches $\{\mathbf{y}_k\}_{k=1}^K$. Instead of naïvely feeding each patch to $\mathrm{CNN}_\mathbf{F}$ individually then performing optimization, we feed the whole noisy image $\mathcal{Y}$ to it, leading to N exemplars images of the same size as $\mathcal{Y}$, denoted as $\{\mathcal{F}_n\}_{n=1}^N$. By doing so, for the $\mathrm{CNN}_\mathbf{F}$ with receptive field size as r, each pixel i on $\mathcal{F}_n$ is influenced by all the pixels j on image $\mathcal{Y}$

if j is in the $r \times r$ neighborhood of i. As a result, for a larger receptive field r, the exemplar $\mathcal{F}_n$ effectively takes into account more *non-local* information for denoising, resembling the notion of non-local means (NLM) in the classic works [5, 10].

With the exemplar images, we simply divide each of them, say, $\mathcal{F}_n$, into K overlapping patches $\mathbf{f}_n^{(k)} \in \mathbb{R}^m$, $1 \le k \le K$. To denoise a patch $\mathbf{y}_k$, we build a graph $\mathcal{G}_k$ with its corresponding N exemplars $\{\mathbf{f}_n^{(k)}\}_{n=1}^N$ in the way described in Section 3.1, leading to the graph Laplacian matrix $\mathbf{L}_k$. Rather than a fully connected graph, we choose the *8-connected pixel* adjacency graph structure, *i.e.*, in the graph $\mathcal{G}_k$, every pixel i is only connected to its 8 neighboring pixels. Hence, the graph Laplacian $\mathbf{L}_k$ is sparse with *fixed* sparsity pattern. The graph Laplacian $\mathbf{L}_k$, together with patch $\mathbf{y}_k$, are passed to the QP solver, which resolves the problem (2) and outputs the denoised patch $\mathbf{x}_k^\star$. By equally aggregating the denoised image patches $\mathbf{x}_k^\star$ ($1 \le k \le K$), we arrive at the denoised image (denoted by $\mathcal{X}^\star$). From spectral graph theory [9], the graph Laplacian regularization layer is always an *adaptive linear low-pass filter*, regardless of the training data.

Apart from the aforementioned procedure, for practical restoration with the graph Laplacian regularization layer, the following ingredients are also adopted.

(i) Generation of μ: in (2), μ trades off the importance between the fidelity term and the graph Laplacian regularizer. To generate the appropriate μ's for regularization, we build a light-weight CNN (denoted as CNN_μ). Particularly, based on the corrupted image $\mathcal{Y}$, it produces a set of $\{\mu_k\}_{k=1}^K$ corresponding to the patches $\{\mathbf{y}_k\}_{k=1}^K$.

(ii) Pre-filtering: in many denoising literature (*e.g.*, [30, 6, 34]), it is popular to perform a pre-filtering operation to the noisy image $\mathcal{Y}$ before optimization. We borrow this idea and implement a pre-filtering step with a light-weight CNN (denoted as $\mathrm{CNN}_{\widehat{y}}$). It operates on image $\mathcal{Y}$ and outputs the filtered image $\widehat{\mathcal{Y}}$. Hence, instead of $\{\mathbf{y}_k\}_{k=1}^K$, we employ the patches of $\widehat{\mathcal{Y}}$, *i.e.*, $\{\widehat{\mathbf{y}}_k\}_{k=1}^K$, in the data term of problem (2).

We call the presented architecture which performs restoration with a graph Laplacian regularization layer *GLRNet*. Figure 2 shows its block diagram, where the graph Laplacian regularization layer is composed of a *graph construction* module generating graph Laplacian matrices, and a *QP solver* producing denoised patches. The denoised image $\mathcal{X}^\star$ is obtained by aggregating the denoised patches. We see that, to achieve denoising, a noisy image first goes through the pre-filtering network $\mathrm{CNN}_{\widehat{y}}$. It is then processed by the graph Laplacian regularization layer, a linear low-pass filter. As a result, our denoising framework is *less*

sensitive to the training data. Moreover, it is *less affected* by the chosen structure of $\mathrm{CNN}_{\widehat{y}}$, as to be seen in Section 4.2.

Since the graph construction involves only elementary functions such as exponentials, powers and arithmetic operations, it is differentiable. Furthermore, from [2] the QP solver is also differentiable with respect to its inputs. Hence, the graph Laplacian regularization layer is fully differentiable, and our denoising pipeline can be *end-to-end* trained. The backward computation of the graph Laplacian regularization layer, including both the graph construction and the QP solver, is provided in the supplementary material.

3.3. Iterative Filtering

To handle the non-Gaussian property of real image noise, we cascade T blocks of GLRNet (each block has a graph Laplacian regularization layer) for effective restoration, leading to our $DeepGLR$ framework. Suppose each noise component removed by a GLRNet follows Gaussian distribution, then the overall noise that is removed forms a mixture of Gaussian. Ideally, it can approximate any distribution arbitrarily well [35]. Moreover, classic literature, *e.g.*, [15, 30, 10], also filter the noisy image iteratively to gradually enhance the image quality. Similar to [43], all the GLRNets in our work have the same structure and share the same parameters. Figure 3 shows the block diagram of DeepGLR. In Figure 3 and the following presentation, we have removed the superscript "$\star$" from $\mathcal{X}^\star$ for simplicity.

To effectively train the proposed DeepGLR framework, we adopt a loss penalizing differences between the recovered image and the ground-truth. Given the noisy image $\mathcal{Y}$, its corresponding ground-truth image $\mathcal{X}^{(\mathrm{gt})}$ and the restoration result $\mathcal{X}_T$, our loss function is defined as the mean-square-error (MSE) between $\mathcal{X}^{(\mathrm{gt})}$ and $\mathcal{X}_T$, *i.e.*,

$$L_{\mathrm{res}}\Big(\mathcal{X}^{(\mathrm{gt})}, \mathcal{X}_T\Big) = \frac{1}{HW} \sum_{i=1}^{H}\sum_{j=1}^{W}\Big(\mathcal{X}^{(\mathrm{gt})}(i,j) - \mathcal{X}_T(i,j)\Big)^2,$$

(5)

where H and W are the height and width of the images, respectively. $\mathcal{X}^{(\mathrm{gt})}(i,j)$ is the (i,j)-th pixel of $\mathcal{X}^{(\mathrm{gt})}$, the same for $\mathcal{X}_T(i,j)$. Note that in our experiments, the restoration loss is only applied to the output of the last cascade $\mathcal{X}_T$, *i.e.*, only the final restoration result is supervised.

For simplicity, we have presented our framework for denoising of 1-channel images. To adapt it for denoising of color images, the first layers of the CNNs are changed to take 3-channel inputs, and $\mathrm{CNN}_{\widehat{y}}$ should output 3 channels. Moreover, in the graph Laplacian regularization layer, the 3 channels share the *same* graph for utilizing inter-color correlation; while the QP solver solves for three separate systems of linear equations then outputs a color image. We choose to work in the YUV color space. During training, the loss function of the 3 channels are computed. We then take the average as the total loss.
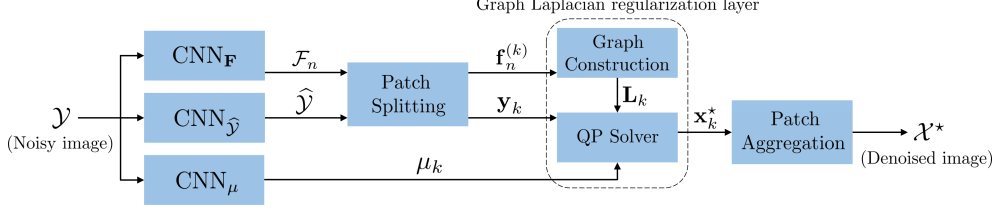
Figure 2. Block diagram of the proposed GLRNet which employs a graph Laplacian regularization layer for image denoising.
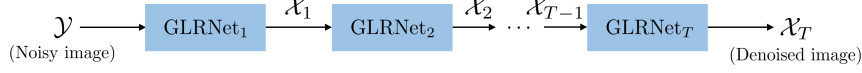

Figure 3. Block diagram of the overall DeepGLR framework.

3.4. Numerical Stability

We hereby analyze the stability of the proposed GLR layer which is indispensable for the stability of the entire framework. Our denoising approach embedding the QP solver into the processing pipeline have numerical stability guarantee. Firstly, the problem (2) essentially boils down to solving a system of linear equations

$$(\mathbf{I} + \mu \mathbf{L}) \mathbf{x}^\star = \mathbf{y}, \tag{6}$$

where $\mathbf{I}$ is an identity matrix. It admits a closed-form solution $\mathbf{x}^\star = (\mathbf{I} + \mu \mathbf{L})^{-1} \mathbf{y}$. Thus, one can interpret $\mathbf{x}^\star$ as a filtered version of noisy input $\mathbf{y}$ with linear filter $(\mathbf{I} + \mu \mathbf{L})^{-1}$. As a combinatorial graph Laplacian, $\mathbf{L}$ is positive semidefinite and its smallest eigenvalue is 0 [39]. Therefore, with $\mu \geq 0$, the matrix $\mathbf{I} + \mu \mathbf{L}$ is always *invertible*, with the smallest eigenvalue as $\lambda_{\min} = 1$. However, the linear system becomes unstable for a numerical solver if $\mathbf{I} + \mu \mathbf{L}$ has a large *condition number* κ—the ratio between the largest and the smallest eigenvalues $\lambda_{\max}/\lambda_{\min}$ for a normal matrix, assuming an l_2-norm [19]. Using eigen-analysis, we have the following theorem regarding κ.

Theorem 1. *The condition number κ of $\mathbf{I} + \mu \mathbf{L}$ satisfies*

$$\kappa \leq 1 + 2\,\mu\,d_{\max}, \tag{7}$$

where $d_{\max}$ is the maximum degree of the vertices in $\mathcal{G}$.

Proof. As discussed, we know $\lambda_{\min} = 1$. By applying the Gershgorin circle theorem [42], $\lambda_{\max}$ can be upper-bounded as follows. First, the i-th Gershgorin disc of $\mathbf{L}$ has radius $r_i = \sum_{j \neq i} |w_{ij}| \leq d_{\max}$, and the center of the disc i for $\mathbf{I} + \mu \mathbf{L}$ is $1 + \mu r_i$. From the Gershgorin circle theorem, the eigenvalues of $\mathbf{I} + \mu \mathbf{L}$ have to reside in the union of all Gershgorin discs. Hence, $\lambda_{\max} \leq \max_i \{1 + 2\,\mu\,r_i\}$, leading to $\kappa = \lambda_{\max} \leq 1 + 2\,\mu\,d_{\max}$. □

Thus, by constraining the value of the weighting parameter μ, we can suppress the condition number κ and ensure a stable denoising filter. Denote the maximum allowable condition number as $\kappa_{\max}$ where we impose $1 + 2\mu d_{\max} \leq \kappa_{\max}$, leading to

$$\mu \leq \frac{\kappa_{\max} - 1}{2d_{\max}} = \mu_{\max}. \tag{8}$$

Hence, if CNN_μ generates a value μ no greater than $\mu_{\max}$, then μ stays unchanged, otherwise it is truncated to $\mu_{\max}$. We empirically set $\kappa_{\max} = 250$ for both training and testing to guarantee the stability of our framework.

4. Experimental Results

Extensive experiments are presented in this section. We first describe our adopted CNN architecture and experimental setup in details then apply our model on real image denoising and validate its robustness. First, it provides satisfactory results when trained with very small amount of data. Moreover, we demonstrate the strong generalization power of our proposal, which outperforms the state-of-the-art approaches by a remarkable margin. We use *peak signal-to-noise ratio* (PSNR) which computes in logarithmic (dB) scale as a measurement for objective evaluation.

4.1. Network Architectures

Our framework does not limit the choices of network architecturesand one has the freedom in designing the specifications of $\mathrm{CNN}_{\mathbf{F}}$, CNN_μ and $\mathrm{CNN}_{\widehat{\mathcal{Y}}}$. In our experiment, we choose the networks shown in Figure 4. Specifically,

(i) $\mathrm{CNN}_{\mathbf{F}}$: To generate exemplars $\{\mathbf{f}_n\}_{n=1}^N$, we adopt the popular hour-glass structure for $\mathrm{CNN}_{\mathbf{F}}$ which has an encoder and a decoder with skip-connections [36]. Similar to [33], we use $N = 3$ exemplars to build the graphs.

(ii) $\mathrm{CNN}_{\widehat{\mathcal{Y}}}$: The pre-filtered image $\widehat{\mathcal{Y}}$ is simply generated by a light-weight CNN with 4 convolution layers using a residual learning structure [18].

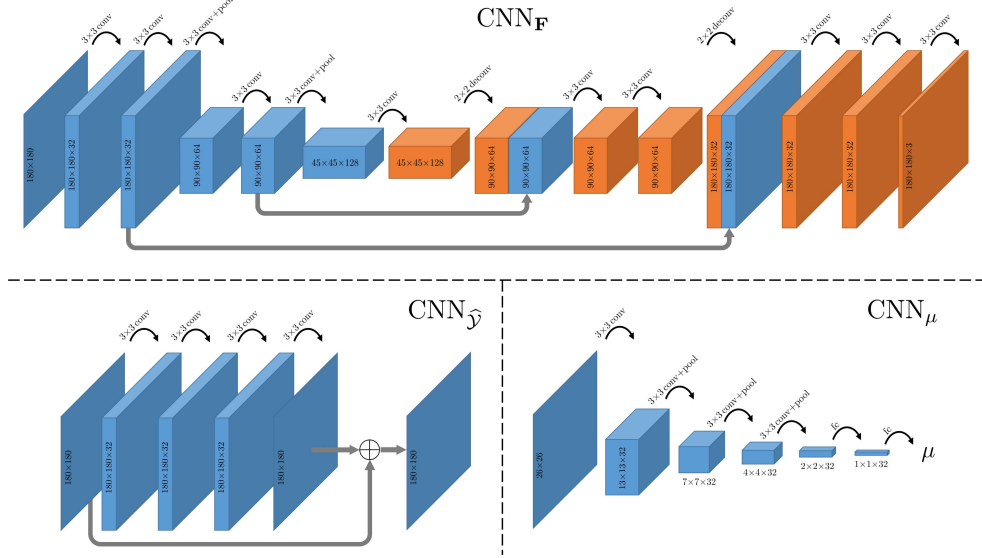(iii) CNN_μ: The weighting parameter μ is estimated on a patch-by-patch basis. Our experiments uses patch size

Figure 4. Network architectures of $\text{CNN}_\mathbf{F}$, $\text{CNN}_{\widehat{y}}$ and CNN_μ in the experiments. Data produced by the decoder of $\text{CNN}_\mathbf{F}$ is colored in orange.

Figure 5. The 10 scenes of the RENOIR dataset [3] used for real image denoising.

of 26×26 for denoising. Hence, starting from a noisy patch, it has undergone 4 convolution layers with 2×2 max pooling and 2 fully-connected layers, leading to the parameter μ.

Except for the last convolution layers of $\text{CNN}_\mathbf{F}$ and $\text{CNN}_{\widehat{y}}$, and the two deconvolution layers of $\text{CNN}_\mathbf{F}$, all the rest network layers shown in Figure 4 are followed by a $\text{ReLU}(\cdot)$ activation function. Note that the input image can have different sizes as long as it is a multiple of 4. For illustration, Figure 4 shows the case when the input is of size 180×180.

4.2. Robust Denoising with Small Training Set

To see the robustness of our proposal, we begin with experimenting DeepGLR with small data. In this experiment, we employ the *RENOIR* [3] dataset, which consists of real low-light noisy images with the corresponding (almost) noise-free versions. Specifically, its subset with 40 scenes collected with a Xiaomi Mi3 smart-phone are used in our experiments. Since some of the scenes have very low intensities while some of the given ground-truth images are still noisy, we remove the scenes whose ground-truths have: (a) average intensities lower than 0.3 (assuming the intensity ranges from 0 to 1); and (b) estimated PSNRs (provided by [3]) lower than $36\,\text{dB}$, leading to 10 valid image pairs. Thumbnails of the images are shown in Figure 5.

We adopt a two-fold cross validation scheme to evaluate the performance of our approach on small dataset. In each of the two trials, we perform training on one fold— only five images—and testing on the other, then measure the performance by the averaging of the results of both trials. The 10 images are randomly split into two folds and we repeat such two-fold cross validation process for five times then the results are averaged. For objective evaluation, peak signal-to-noise ratio (PSNR) and structural similarity (SSIM) [44] are employed. During the training phase, the noisy images, accompanied with their noise-free versions, are fed to the network for training. For both training and testing, the overlapping patches are of size 26×26, *i.e.*, $m = 26^2 = 676$, where neighboring patches are of a stride 22 apart. We let the batch size be 4 and the model is trained for 200 epochs. A multi-step learning rate decay policy, with values $[1, 0.5, 0.1, 0.05, 0.01, 0.005] \times 10^{-3}$, are used, where the learning rate decreases at the beginning of epochs $[2, 5, 20, 50, 150]$. We implement the network with TensorFlow [1] on an Nvidia GeForce GTX Titan X GPU. Note that the QP solver is implemented with the TensorFlow layer, *i.e.*, `matrix_solve_ls`, for solving a system of linear equations in the least squares sense.

Our DeepGLR is compared with the following approaches: (a) CBM3D dedicated for Gaussian noise removal on color image [10][1]; (b) MC-WNNM designed for real image noise removal [46]; (c) Noise clinic [25] also

---

[1]For testing with CBM3D, we estimate the equivalent noise variances using the ground-truth and the noisy images.

(a) Ground-truth   (b) Noisy   (c) CBM3D   (d) MC-WNNM   (e) Noise Clinic

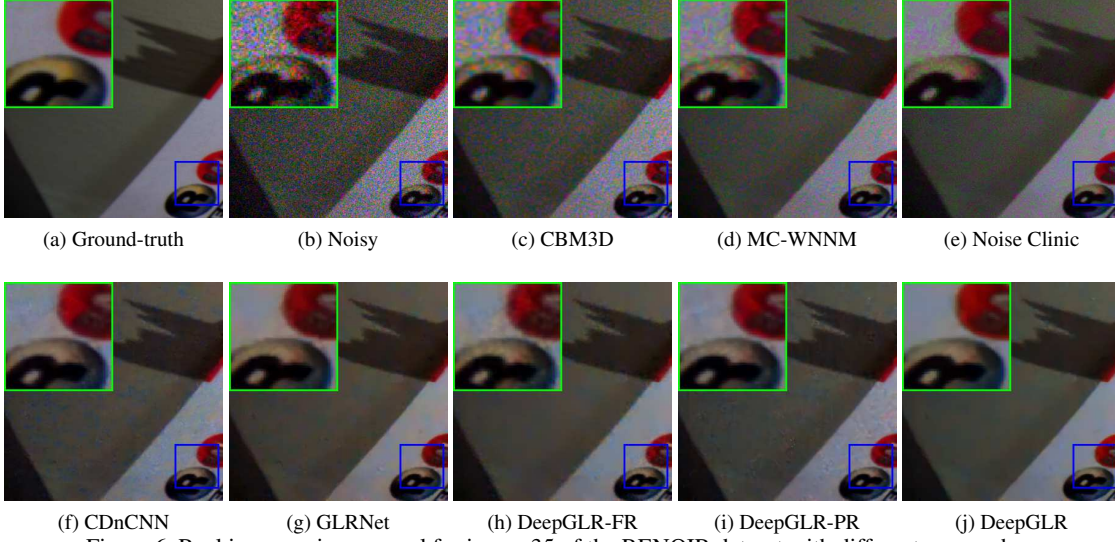(f) CDnCNN   (g) GLRNet   (h) DeepGLR-FR   (i) DeepGLR-PR   (j) DeepGLR

Figure 6. Real image noise removal for image 35 of the RENOIR dataset with different approaches.

Table 1. Evaluation of different methods for real image denoising. The best results for each metric, except for those tested on the training set, are highlighted in boldface.

Metric	Noisy	Method						
		CBM3D	MC-WNNM	Noise Clinic	CDnCNN (Train)	CDnCNN (Test)	DeepGLR (Train)	DeepGLR (Test)
PSNR	20.63	26.08	26.23	27.43	34.82	32.79	34.28	**32.96**
SSIM	0.3081	0.6727	0.6294	0.6040	0.8852	0.8583	0.8795	**0.8634**

designed for real image noise removal; and (d) CDnCNN [48], a data-driven approach trained with the same dataset as ours. Evaluation results are shown in Table 1, where DeepGLR outperforms competing schemes by a range of 0.17–6.88 dB. More visual results are demonstrated in Figure 6, where competing schemes fail to fully remove the noise, while DeepGLR is more satisfactory. To see the gap between training and testing, performance on the training set is also measured as shown in columns CDnCNN (Train) and DeepGLR (Train) in Table 1, where CDnCNN excels in training set but not in testing set indicating a strong overfitting. This is because:

(i) Only *5 images* are available for training in this experiment, letting CDnCNN strongly *overfit* to the training data. However, our DeepGLR is less sensitive to the deficiency of the training data.

(ii) While CDnCNN is most suitable for Gaussian noise removal (as stated in [48]), our DeepGLR adaptively learns the suitable graphs to low-pass filter the real noisy image, which weakens the impact of the complex real noise statistics.

To better understand DeepGLR, we also consider several of its variant: (a) The pre-filtering network $CNN_{\widehat{y}}$ is removed, we call the resulting method DeepGLR-PR ("PR" stands for pre-filter removed); (b) $CNN_{\widehat{y}}$ is replaced by CDnCNN, the resulting method is called DeepGLR-PC ("PC" stands for pre-filter with CDnCNN); (c) $CNN_{\mathbf{F}}$ is removed, and directly use the output of $CNN_{\widehat{y}}$ as exemplars for graph construction, this scheme is referred to as DeepGLR-FR ("FR" stands for $CNN_{\mathbf{F}}$ removed); (d) GLRNet. Evaluations are provided in Table 2, where DeepGLR-PC provides similar performance as DeepGLR, suggesting the GLR layer can perform effective denoising, and adding extra layers to $CNN_{\mathbf{F}}$ is of little use. Moreover, DeepGLR-PC (33.03 dB) can be regarded as a CDnCNN module (32.79 dB) with GLR as the post-processing, indicating that GLR boosts CDnCNN's performance. Apart from DeepGLR-PC, the others provide less satisfactory results, which is consistent with results in Figure 6. Without the module for exemplar learning (DeepGLR-FR), DeepGLR cannot capture the underlying image structure; without pre-filtering, the GLR layer has limited effect (DeepGLR-PR); without iterative filtering, one GLRNet alone cannot fully remove real noise with complicated statistics. In light of this, DeepGLR stands as a composite of modules, each playing an irreplaceable role.

4.3. Cross-Domain Generalization

We hereby evaluate the robustness of our approach in terms of its *cross-domain generalization ability*. Specifically, we evaluate on the RENOIR dataset with DeepGLR and CDnCNN trained for AWGN blind denoising.
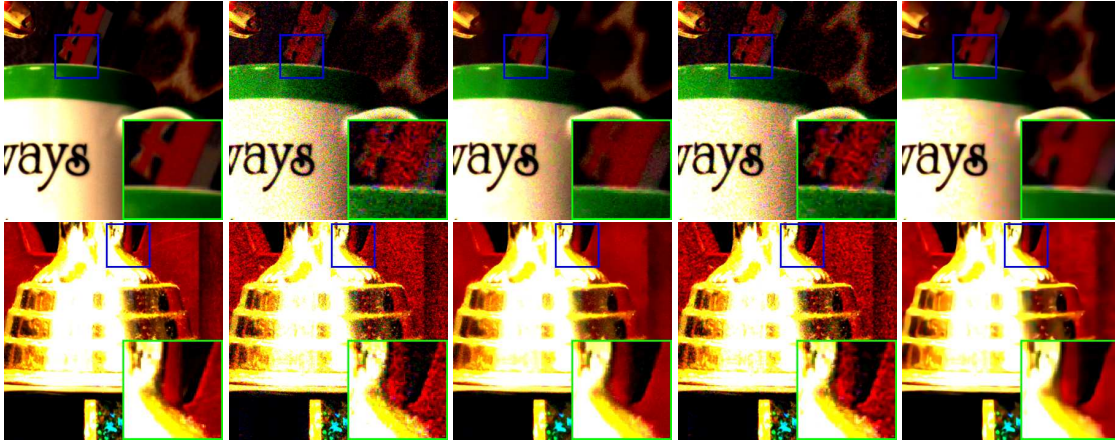
| (a) Ground-truth | (b) Noisy | (c) Noise Clinic | (d) CDnCNN | (e) DeepGLR |

Figure 7. The proposed DeepGLR trained for AWGN denoising generalizes well to real image denoising.

Table 2. Evaluation of different variants of DeepGLR for real image denoising. The best two results for each metric are highlighted in boldface.

| Metric | DeepGLR | Variant of DeepGLR | | | |
		DeepGLR -PR	DeepGLR -PC	DeepGLR -FR	GLRNet
PSNR	**32.9593**	32.33	**33.025**	32.51	32.50
SSIM	**0.8634**	0.8534	**0.8637**	0.8552	0.8599

During training, we adopts the same training dataset as CDnCNN, *i.e.*, 432 images from BSDS500, and covers the noise level range $\sigma \in [0, 55]$. The same network structure, optimizer and learning rate settings are used for AWGN denoising as for real noise removal detailed in Section 4.2.

We first evaluate AWGN blind denoising where the results, in terms of PSNR and SSIM, are shown in Table 3. The rest 68 images from BSDS500 are adopted for evaluation with noise standard deviations 15, 25, 50 with CBM3D as the baseline method. DeepGLR has results *on par with* CDnCNN, both of them surpasses CBM3D by a large margin. The results of OGLR [33] are also shown in Table 3.

More importantly, we evaluate the performance of DeepGLR and CDnCNN on RENOIR dataset to investigate their **cross-domain generalization** ability. For comparison, we include noise clinic [25] designed for real noise removal as a baseline method. Objective performance, in terms of PSNR and SSIM, are listed in Table 4. We see that, DeepGLR has a PSNR performance of 30.10 dB, *outperforming CDnCNN by 5.74 dB, and noise clinic by 1.87 dB*. Nevertheless, CDnCNN provides better results in Gaussian noise removal as shown in Table 3. This indicates that CDnCNN is **strongly overfitted** to the case of Gaussian noise removal and fails to generalize to real noise, while DeepGLR provides satisfactory denoising results.

Subjective results demonstrated in Figure 7 show the denoising results of two image fragments from the RENOIR dataset. Noise clinic still has noticable noise unremoved, and CDnCNN almost preserves most noise. In contrast, our DeepGLR provides the best visual quality, removing the noise while preserving the sharp edge details. The strong domain generalization stems from robust exemplar learning module in capturing the intrinsic image structure with presence of complex noise, boosting the denoising performance through the GLR layer in recovering the clean image.

Table 3. Average PSNR (dB) and SSIM values for Gaussian noise removal.

| Noise | Method (PSNR / SSIM) | | | |
	CBM3D	CDnCNN	OGLR	DeepGLR
15	33.49 / 0.9216	33.80 / 0.9268	33.52 / 0.9198	33.65 / 0.9259
25	30.68 / 0.8675	31.13 / 0.8799	30.79 / 0.8661	31.03 / 0.8797
50	27.35 / 0.7627	27.91 / 0.7886	27.84 / 0.7755	27.86 / 0.7924

Table 4. Evaluation of cross-domain generalization for real image denoising. The best results are highlighted in boldface.

| Metric | Noisy | Method | | |
		Noise Clinic	CDnCNN	DeepGLR
PSNR	20.36	27.43	24.36	**30.10**
SSIM	0.1823	0.6040	0.5206	**0.8028**

5. Conclusion

In this work, we incorporate graph Laplacian regularization into a deep learning framework for real image noise removal. Given a corrupted image, it is first fed to a CNN, then neighborhood graphs are constructed from the CNN outputs on a patch-by-patch basis. The graph construction and the denoising process is fully differentiable, hence the overall pipeline can be end-to-end trained. We demonstrate the robustness of the proposed framework—DeepGLR—for real noise removal, from two different aspects. Firstly, it demonstrates higher immunity to overfitting while training with small dataset. Moreover, it manifests strong cross-domain generalization ability when training and testing data have different statistics.

References

[1] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard, et al. Tensorflow: A system for large-scale machine learning. In *OSDI*, volume 16, pages 265–283, 2016. 6

[2] B. Amos and J. Z. Kolter. OptNet: Differentiable optimization as a layer in neural networks. In *Proceedings of International Conference on Machine Learning (ICML)*, pages 136–145. PMLR, 2017. 3, 4

[3] J. Anaya and A. Barbu. Renoir—A dataset for real low-light noise image reduction. *Journal of Visual Communication and Image Representation*, 51:144 – 154, 2018. 6

[4] J. T. Barron and B. Poole. The fast bilateral solver. In *European Conference on Computer Vision*, pages 617–632. Springer, 2016. 3

[5] A. Buades, B. Coll, and J.-M. Morel. A non-local algorithm for image denoising. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 2, pages 60–65. IEEE, 2005. 1, 2, 4

[6] P. Chatterjee and P. Milanfar. Patch-based near-optimal image denoising. *IEEE Transactions on Image Processing*, 21(4):1635–1649, 2012. 4

[7] C. Chen, Q. Chen, J. Xu, and V. Koltun. Learning to see in the dark. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3291–3300, June 2018. 1, 2

[8] C. Chen, Z. Xiong, X. Tian, and F. Wu. Deep boosting for image denoising. In *The European Conference on Computer Vision (ECCV)*, September 2018. 2

[9] F. R. Chung and F. C. Graham. *Spectral graph theory*. Number 92. American Mathematical Soc., 1997. 4

[10] K. Dabov, A. Foi, V. Katkovnik, and K. Egiazarian. Image denoising by sparse 3-D transform-domain collaborative filtering. *IEEE Transactions on image processing*, 16(8):2080–2095, 2007. 2, 4, 6

[11] M. Defferrard, X. Bresson, and P. Vandergheynst. Convolutional neural networks on graphs with fast localized spectral filtering. In *Advances in Neural Information Processing Systems*, pages 3844–3852, 2016. 3

[12] C. Dong, C. C. Loy, K. He, and X. Tang. Learning a deep convolutional network for image super-resolution. In *European Conference on Computer Vision*, pages 184–199. Springer, 2014. 2

[13] W. Dong, P. Wang, W. Yin, G. Shi, F. Wu, and X. Lu. Denoising prior driven deep neural network for image restoration. *arXiv preprint arXiv:1801.06756*, 2018. 1

[14] H. E. Egilmez, E. Pavez, and A. Ortega. Graph learning from data under Laplacian and structural constraints. *IEEE Journal of Selected Topics in Signal Processing*, 11(6):825–841, 2017. 3

[15] M. Elad and M. Aharon. Image denoising via sparse and redundant representations over learned dictionaries. *IEEE Transactions on Image processing*, 15(12):3736–3745, 2006. 1, 2, 4

[16] A. Elmoataz, O. Lezoray, and S. Bougleux. Nonlocal discrete regularization on weighted graphs: A framework for image and manifold processing. *IEEE Transactions on Image Processing*, 17(7):1047–1060, 2008. 2, 3

[17] G. Gilboa and S. Osher. Nonlocal linear image regularization and supervised segmentation. *Multiscale Modeling & Simulation*, 6(2):595–630, 2007. 2

[18] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016. 5

[19] R. A. Horn and C. R. Johnson. *Matrix analysis*. Cambridge University Press, 1990. 5

[20] W. Hu, G. Cheung, and M. Kazui. Graph-based dequantization of block-compressed piecewise smooth images. *IEEE Signal Processing Letters*, 23(2):242–246, 2016. 2, 3

[21] V. Jain and S. Seung. Natural image denoising with convolutional networks. In *Advances in Neural Information Processing Systems*, pages 769–776, 2009. 2

[22] A. Kheradmand and P. Milanfar. A general framework for regularized, similarity-based image restoration. *IEEE Transactions on Image Processing*, 23(12):5136–5151, 2014. 3

[23] T. N. Kipf and M. Welling. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations (ICLR)*, 2017. 3

[24] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems*, pages 1097–1105, 2012. 2

[25] M. Lebrun, M. Colom, and J.-M. Morel. The noise clinic: A blind image denoising algorithm. *Image Processing On Line*, 5:1–54, 2015. 6, 8

[26] S. Lefkimmiatis. Universal denoising networks: A novel CNN architecture for image denoising. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3204–3213, 2018. 2

[27] X. Liu, D. Zhai, D. Zhao, G. Zhai, and W. Gao. Progressive image denoising through hybrid graph Laplacian regularization: A unified framework. *IEEE Transactions on image processing*, 23(4):1491–1503, 2014. 2, 3

[28] X. Mao, C. Shen, and Y.-B. Yang. Image restoration using very deep convolutional encoder-decoder networks with symmetric skip connections. In *Advances in Neural Information Processing Systems*, pages 2802–2810, 2016. 2

[29] M. T. McCann, K. H. Jin, and M. Unser. Convolutional neural networks for inverse problems in imaging: A review. *IEEE Signal Processing Magazine*, 34(6):85–95, 2017. 1

[30] P. Milanfar. A tour of modern image filtering: New insights and methods, both practical and theoretical. *IEEE Signal Processing Magazine*, 30(1):106–128, 2013. 2, 4

[31] S. Osher, Z. Shi, and W. Zhu. Low dimensional manifold model for image processing. *SIAM Journal on Imaging Sciences*, 10(4):1669–1690, 2017. 3

[32] W. Ouyang and X. Wang. Joint deep learning for pedestrian detection. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 2056–2063, 2013. 2

[33] J. Pang and G. Cheung. Graph Laplacian regularization for image denoising: Analysis in the continuous domain. *IEEE*

Transactions on Image Processing, 26(4):1770–1785, 2017. 1, 2, 3, 5, 8

[34] J. Pang, G. Cheung, A. Ortega, and O. C. Au. Optimal graph Laplacian regularization for natural image denoising. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 2294–2298, 2015. 4

[35] D. Reynolds. Gaussian mixture models. *Encyclopedia of biometrics*, pages 827–832, 2015. 4

[36] O. Ronneberger, P. Fischer, and T. Brox. U-Net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 234–241. Springer, 2015. 5

[37] L. I. Rudin, S. Osher, and E. Fatemi. Nonlinear total variation based noise removal algorithms. *Physica D: nonlinear phenomena*, 60(1-4):259–268, 1992. 1

[38] X. Shen, X. Tao, H. Gao, C. Zhou, and J. Jia. Deep automatic portrait matting. In *European Conference on Computer Vision*, pages 92–107. Springer, 2016. 3

[39] D. I. Shuman, S. K. Narang, P. Frossard, A. Ortega, and P. Vandergheynst. The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains. *IEEE Signal Processing Magazine*, 30(3):83–98, 2013. 3, 5

[40] Y. Tai, J. Yang, X. Liu, and C. Xu. Memnet: A persistent memory network for image restoration. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4539–4547, 2017. 1, 2

[41] S. C. Turaga, J. F. Murray, V. Jain, F. Roth, M. Helmstaedter, K. Briggman, W. Denk, and H. S. Seung. Convolutional networks can learn to generate affinity graphs for image segmentation. *Neural Computation*, 22(2):511–538, 2010. 3

[42] R. S. Varga. *Geršgorin and his circles*, volume 36. Springer Science & Business Media, 2010. 5

[43] R. Vemulapalli, O. Tuzel, and M.-Y. Liu. Deep Gaussian conditional random field network: A model-based deep network for discriminative denoising. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4801–4809, 2016. 1, 2, 4

[44] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli. Image quality assessment: From error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13(4):600–612, 2004. 6

[45] J. Xu, L. Zhang, and D. Zhang. A trilateral weighted sparse coding scheme for real-world image denoising. In *European Conference on Computer Vision (ECCV)*, September 2018. 2

[46] J. Xu, L. Zhang, D. Zhang, and X. Feng. Multi-channel weighted nuclear norm minimization for real color image denoising. In *The IEEE International Conference on Computer Vision (ICCV)*, Oct 2017. 1, 6

[47] L. Xu, J. S. Ren, C. Liu, and J. Jia. Deep convolutional neural network for image deconvolution. In *Advances in Neural Information Processing Systems*, pages 1790–1798, 2014. 2

[48] K. Zhang, W. Zuo, Y. Chen, D. Meng, and L. Zhang. Beyond a Gaussian denoiser: Residual learning of deep CNN for image denoising. *IEEE Transactions on Image Processing*, 26(7):3142–3155, 2017. 1, 2, 7

[49] K. Zhang, W. Zuo, and L. Zhang. Ffdnet: Toward a fast and flexible solution for cnn based image denoising. *IEEE Transactions on Image Processing*, 2018. 2

[50] F. Zhu, G. Chen, and P.-A. Heng. From noise modeling to blind image denoising. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 420–429, 2016. 1