

Supplementary Material for “Deep Graph Laplacian Regularization for Robust Denoising of Real Images”

Jin Zeng^{1*} Jiahao Pang^{1*} Wenxiu Sun¹ Gene Cheung²
¹SenseTime Research ²Department of EECS, York University
 {zengjin, pangjiahao, sunwenxiu}@sensetime.com, genec@yorku.ca

1. Introduction

In this supplementary material, we first derive the backward computation of the proposed graph Laplacian regularization layer. We then present additional visual comparisons of image denoising, including both the cases of real image denoising and vanilla Gaussian denoising, with our DeepGLR framework.

2. Backpropagation of the Graph Laplacian Regularization Loss

We hereby derive the backward computation of the proposed graph Laplacian regularization layer which consists of the graph construction module and the QP solver. Suppose for a noisy patch \mathbf{y} , its corresponding recovered patch is \mathbf{x} while the underlying ground-truth is $\mathbf{x}^{(\text{gt})}$, where $\mathbf{x}, \mathbf{x}^{(\text{gt})}, \mathbf{y} \in \mathbb{R}^m$. For simplicity, we consider a loss function defined on a patch basis, which computes the weighted Euclidean distance between \mathbf{x} and the ground-truth $\mathbf{x}^{(\text{gt})}$, *i.e.*,

$$\begin{aligned} e &= \frac{1}{2} \left\| \mathbf{C} (\mathbf{x} - \mathbf{x}^{(\text{gt})}) \right\|_2^2 \\ &= \frac{1}{2} (\mathbf{x} - \mathbf{x}^{(\text{gt})})^\top \mathbf{C}^\top \mathbf{C} (\mathbf{x} - \mathbf{x}^{(\text{gt})}), \end{aligned} \quad (1)$$

where $\mathbf{C} = \text{diag}\{c_1, c_2, \dots, c_m\}$ is a diagonal matrix and $c_i \geq 0$ represents the weight of the i -th pixel. Consequently, the loss function L_{res} in our paper can be regarded as the summation of a series of patch-based loss (1) with the corresponding matrices \mathbf{C} 's.

QP solver: we first consider the backward pass of the QP solver, *i.e.*, we derive the error propagation of the weighting parameter μ and the graph Laplacian matrix \mathbf{L} . From [1],

we have

$$\begin{aligned} \frac{\partial e}{\partial \mu} &= \left(\frac{\partial e}{\partial \mathbf{x}} \right)^\top \cdot \frac{\partial \mathbf{x}}{\partial \mu} \\ &= \left(\mathbf{C}^\top \mathbf{C} (\mathbf{x} - \mathbf{x}^{(\text{gt})}) \right)^\top \cdot \frac{\partial \left((\mathbf{I} + \mu \mathbf{L})^{-1} \cdot \mathbf{y} \right)}{\partial \mu} \\ &= - \left(\mathbf{C}^\top \mathbf{C} (\mathbf{x} - \mathbf{x}^{(\text{gt})}) \right)^\top (\mathbf{I} + \mu \mathbf{L})^{-1} \mathbf{L} (\mathbf{I} + \mu \mathbf{L})^{-1} \mathbf{y}. \end{aligned} \quad (2)$$

We denote $\delta_i \in \mathbb{R}^m$ as the indication vector whose i -th entry is 1 while the rest are zeros, then

$$\begin{aligned} \frac{\partial e}{\partial \mathbf{L}} &= \sum_{i=1}^m \frac{\partial e}{\partial \mathbf{x}(i)} \cdot \frac{\partial \mathbf{x}(i)}{\partial \mathbf{L}} \\ &= \frac{1}{2} \sum_{i=1}^m \frac{\partial (\mathbf{x}(i) - \mathbf{x}^{(\text{gt})}(i))^2}{\mathbf{x}(i)} \cdot \frac{\partial \left(\delta_i^\top \cdot (\mathbf{I} + \mu \mathbf{L})^{-1} \cdot \mathbf{y} \right)}{\partial (\mathbf{I} + \mu \mathbf{L})} \\ &= - \sum_{i=1}^m \mu c_i^2 \left(\mathbf{x}(i) - \mathbf{x}^{(\text{gt})}(i) \right) \\ &\quad \cdot (\mathbf{I} + \mu \mathbf{L})^{-\top} \delta_i \mathbf{y}^\top (\mathbf{I} + \mu \mathbf{L})^{-\top}, \end{aligned} \quad (3)$$

where $\mathbf{x}(i)$ is the i -th entry of \mathbf{x} , the same for $\mathbf{x}^{(\text{gt})}(i)$.

Graph construction: we hereby derive the partial derivative of the graph Laplacian matrix \mathbf{L} with respect to the i -th entry of the exemplar \mathbf{f}_n where $1 \leq i \leq m$, $1 \leq n \leq N$. From the definition of the graph Laplacian matrix and (3)(4) of our paper,

$$\begin{aligned} \frac{\partial \mathbf{L}}{\partial \mathbf{f}_n(i)} &= \sum_{j \in \mathcal{N}(i)} \frac{\partial \mathbf{L}}{\partial w_{ij}} \cdot \frac{\partial w_{ij}}{\partial \mathbf{f}_n(i)} \\ &= \sum_{j \in \mathcal{N}(i)} \left(\delta_i \delta_i^\top + \delta_j \delta_j^\top - \delta_i \delta_j^\top - \delta_j \delta_i^\top \right) \cdot \left(\frac{w_{ij}}{\epsilon^2} (\mathbf{f}_n(j) - \mathbf{f}_n(i)) \right), \end{aligned} \quad (4)$$

where $\mathcal{N}(i)$ denotes the 8 neighboring pixels of pixel i .

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

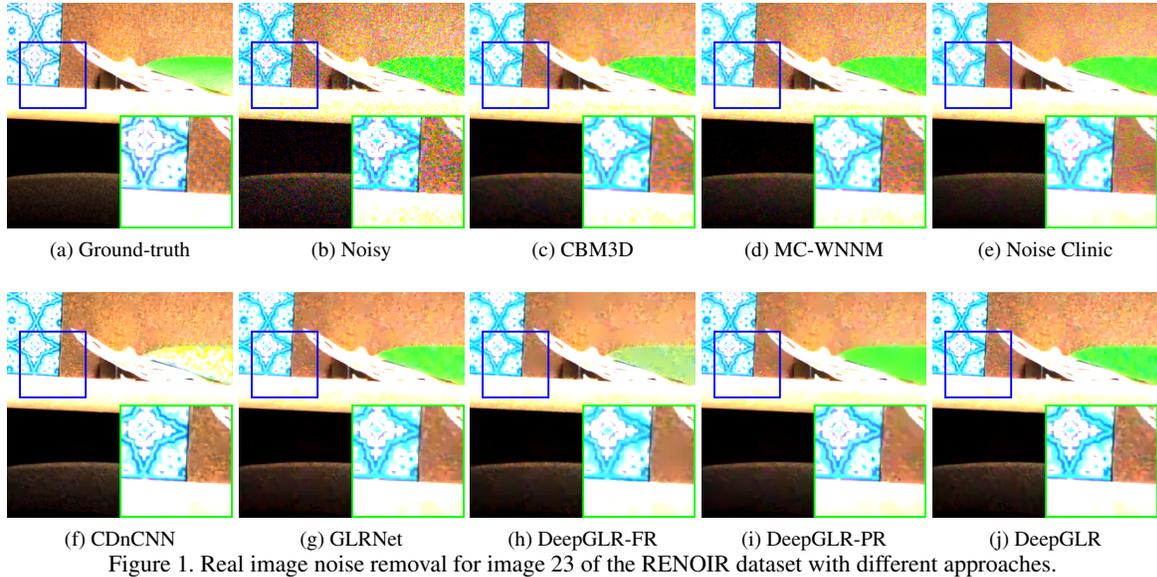


Figure 1. Real image noise removal for image 23 of the RENOIR dataset with different approaches.

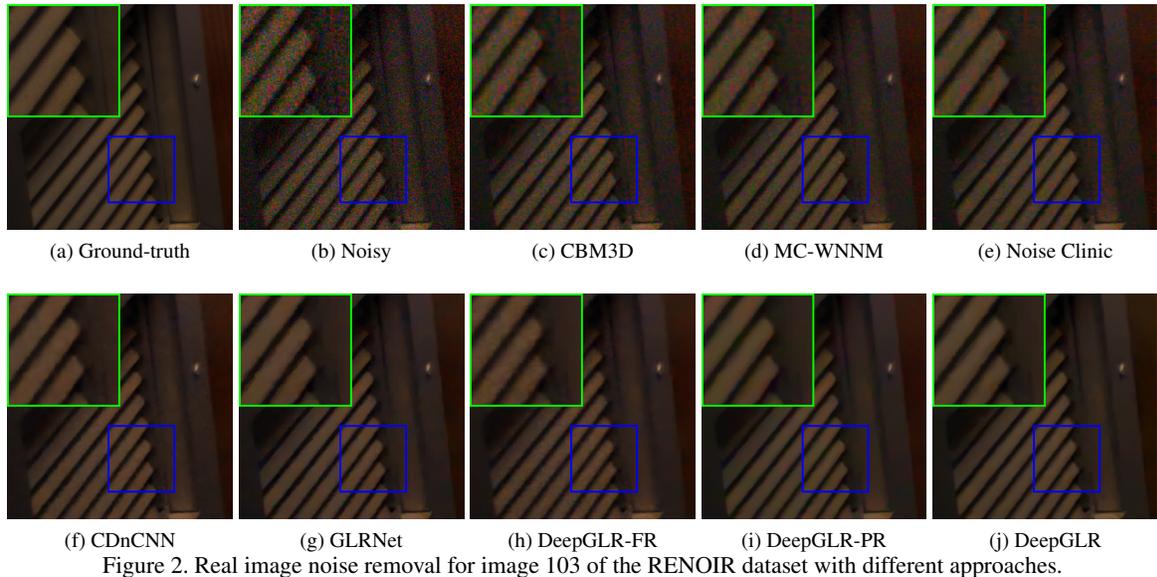


Figure 2. Real image noise removal for image 103 of the RENOIR dataset with different approaches.

3. More Visual Results

In this section, we present more visual results applying DeepGLR to image denoising, with the same settings described in our paper. In all the figures, we enlarge the block in the blue rectangle and display it in the green rectangle for a better view.

Figure 1 and Figure 2 show visual results on real image denoising trained on small dataset as discussed in Section 4.2 in the paper. Competing schemes exhibit noticeable noise, while the proposed DeepGLR generates clean and sharp results. Meanwhile, the variants of DeepGLR does not show competitive results as DeepGLR, indicating

that any component of the our proposed architecture is irreplaceable.

In Figure 3, we additionally show results of real image denoising with models trained for Gaussian noise removal in order to test cross-domain generalization as discussed in Section 4.3 in the paper. Meanwhile, Figure 4 show visual results on Gaussian noise removal.

We see that CDnCNN trained on AWGN dataset fails at real noise removal as shown in Figure 3, indicating strong overfit to AWGN denoising. On the contrary, DeepGLR is still able to capture underlying signal structure and remove the noise with high quality. Moreover, as shown in Figure 4, DeepGLR demonstrates competitive denoising per-

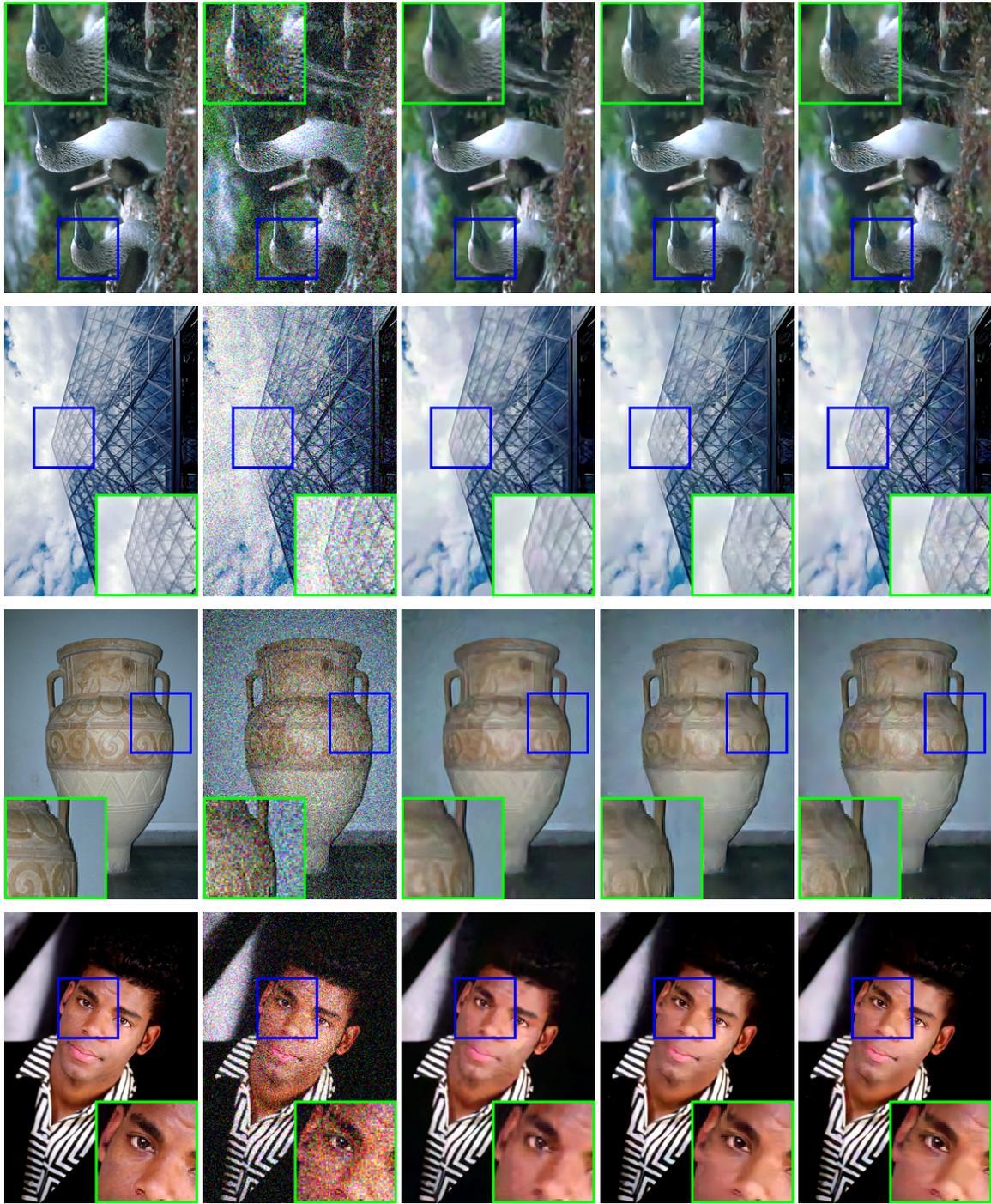


(a) Ground-truth (b) Noisy (c) Noise Clinic (d) CDnCNN (e) DeepGLR
 Figure 3. The proposed DeepGLR trained for AWGN denoising generalizes well to real image denoising.

formance as CDnCNN for AWGN. We see that DeepGLR well preserves the sharp details without over-sharpening, while DnCNN shows less sharp edges and loss of details in the first three images in Figure 4, and generates extra sharp features in the last image. Note that images in the first two rows are rotated counterclockwise by 90° for a compact-yet-clear layout.

References

- [1] K. B. Petersen, M. S. Pedersen, et al. The matrix cookbook. *Technical University of Denmark*, 7(15):510, 2008. 1



(a) Ground-truth

(b) Noisy

(c) CBM3D

(d) CDnCNN

(e) DeepGLR

Figure 4. Subjective evaluation of AWGN denoising on BSD68 Dataset.