Learning Fully Convolutional Networks for Iterative Non-blind Deconvolution

Jiawei Zhang[13*] Jinshan Pan[2] Wei-Sheng Lai[3] Rynson W.H. Lau[1] Ming-Hsuan Yang[3]
Department of Computer Science, City University of Hong Kong[1]
School of Mathematical Sciences, Dalian University of Technology[2]
Electrical Engineering and Computer Science, University of California, Merced[3]

Abstract

In this paper, we propose a fully convolutional network for iterative non-blind deconvolution. We decompose the non-blind deconvolution problem into image denoising and image deconvolution. We train a FCNN to remove noise in the gradient domain and use the learned gradients to guide the image deconvolution step. In contrast to the existing deep neural network based methods, we iteratively deconvolve the blurred images in a multi-stage framework. The proposed method is able to learn an adaptive image prior, which keeps both local (details) and global (structures) information. Both quantitative and qualitative evaluations on the benchmark datasets demonstrate that the proposed method performs favorably against state-of-the-art algorithms in terms of quality and speed.

1. Introduction

Single image non-blind deconvolution aims to recover a sharp latent image given a blurred image and the blur kernel. The community has made active research effort on this classical problem in the last decade. Assuming the camera motion is spatially invariant, a blurred image y can be modeled as a convolution using a blur kernel k and a latent image x:

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

where n is additive noise and $*$ is the convolution operator. In non-blind deconvolution, we solve x from y and k. This is an ill-posed problem since the noise is unknown.

Conventional approaches, such as the Richardson-Lucy deconvolution [20] and the Wiener filter [33], suffer from serious ringing artifacts and thus are less effective to deal with large motion and outliers. Several methods focus on developing effective image priors for image restoration, including Hyper-Laplacian priors [14, 15], non-local means [2], fields of experts [23, 24, 26, 27], patch-based priors [30, 39] and shrinkage fields [25]. However, these image priors are heavily based on the empirical statistics of natural images, and they typically lead to highly non-convex optimization problems. Meanwhile, most of the aforementioned methods have high computational costs.

Recently, deep neural networks have been applied to image restoration [28, 35]. However, these methods need to re-train the network for different blur kernels, which is not practical in real-world scenarios.

Different from existing methods, we propose an iterative FCNN for non-blind deconvolution, which is able to automatically learn effective image priors and does not need to re-train the network for different blur kernels. The proposed method decomposes the non-blind deconvolution into two steps: image denoising and image deconvolution. In the image denoising step, we train a FCNN to remove noise and outliers in the *gradient* domain. The learned image gradients are treated as image priors to guide image deconvolution. In the image deconvolution step, we concatenate a deconvolution module at the end of the FCNN to remove the blur from the input image. We cascade the FCNN into a multi-stage architecture to deconvolve blurred images in an iterative manner. The proposed FCNN adaptively learns effective image priors to preserve image details and structures. In order to effectively suppress ringing artifacts and noise in the smooth regions, we propose to optimize the FCNN with a robust L_1 loss function instead of a commonly used L_2 loss function. In addition, we optimize the hyper-parameters in the deconvolution modules. Extensive evaluations on the benchmark datasets demonstrate that the proposed method performs favorably against state-of-the-art algorithms in terms of quality and speed.

2. Related Work

Non-blind deconvolution has been studied extensively and numerous algorithms have been proposed. In this section, we discuss the most relevant algorithms and put this work in proper context.

Since non-blind deblurring is an ill-posed problem, it requires some assumptions or prior knowledge to constrain the solution space. The early approaches, *e.g.*, Wiener deconvolution [33], assume that the value of every pixel

*email: zhjw1988@gmail.com

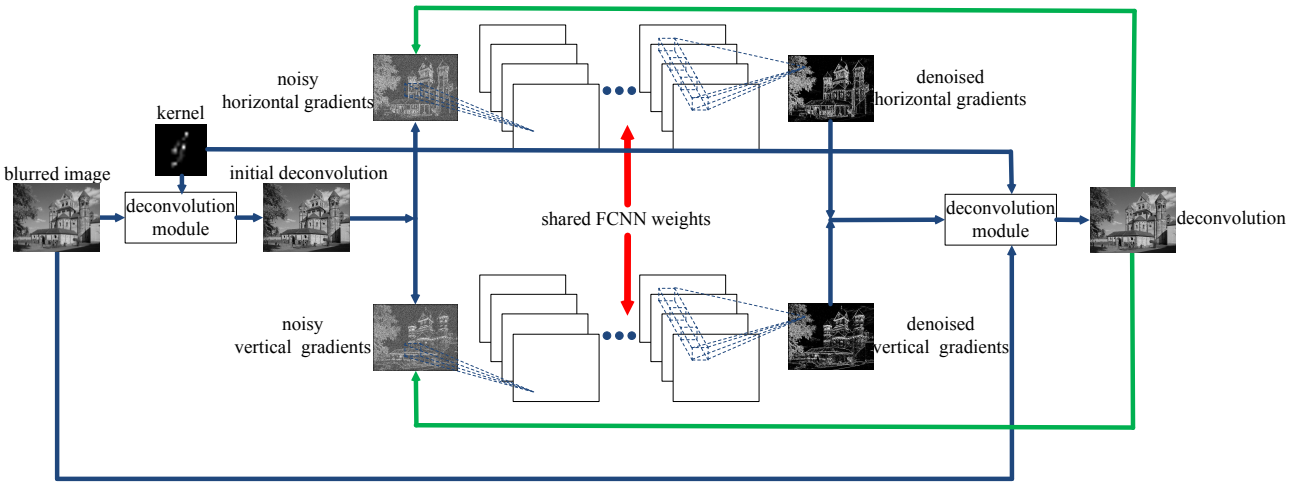


Figure 1. The network structure. Our network first deconvolves the blurry input image by the deconvolution module and then performs convolutions to the vertical and horizontal gradients to generate the results with less noise. After that, the deconvolution module is applied to the denoised gradients to generate the clear image. The gradients of the clear image will then become the inputs of convolutions in the next iteration. We iteratively perform the above steps three times and estimate the final deblurred image. See text for more details.

should follow Gaussian distribution. However, this assumption does not hold for natural images as the distribution of real-world image gradient is heavy-tailed. To develop an image prior that fits the heavy-tailed distribution of natural images, the Hyper-Laplacian prior is proposed [15]. As solving the image restoration with the Hyper-Laplacian prior is time-consuming, Krishnan and Fergus [14] propose an efficient algorithm based on a half-quadratic splitting method.

To learn good priors for image restoration, Roth and Black [23] learn a group of fields of experts (FOEs) to fit the heavy-tailed distribution of natural images. The FOE framework is further extended by [25, 26]. However, methods with fields of experts usually lead to complex optimization problems. Solving these problems are usually time-consuming.

The Gaussian Mixture Model (GMM) has also been developed to fit the heavy-tailed distribution of natural image gradient. Fergus *et al*. [9] use a mixture of Gaussians to learn an image gradient prior via variational Bayesian inference. Zoran and Weiss [39] analyze the image priors in image restoration and propose a patch based prior based on GMM. This work is further extended by Sun *et al*. [30]. Although good results have been achieved, solving these methods needs heavy computation loads.

Recently deep learning has been used for low-level image processing such as denoising [3, 8, 10, 34], super-resolution [6, 11, 12, 21, 22, 32, 37], dehazing [19] and edge-preserving filtering [16, 17, 36]. For non-blind deblurring, Schuler *et al*. [28] develop a multi-layer perceptron (MLP) approach to remove noise and artifacts which are produced by the deconvolution process. Xu *et al*. [35] use a deep CNN to restore images with outliers. This method uses singular value decomposition (SVD) to reduce the number of parameters in the network. However, it needs to fine-tune the network for every kernel as it uses SVD of the pseudo inverse kernel as the network initialization. Different from existing CNN-based method, we develop an effective iterative FCNN for non-blind deconvolution. We cascade the FCNN into a multi-stage architecture to deconvolve blurred images in an iterative manner to preserve the details of the restored images. Moreover, our method does not retain model for each blur kernel.

3. Proposed Algorithm

In this section, we present our algorithm that learns effective image priors for non-blind image deconvolution. We first review half-quadratic optimization in image restoration and then introduce our method.

3.1. Motivation

The half-quadratic splitting framework has been widely used in non-blind deblurring methods [14, 25, 26, 39]. We first review this method in image restoration and then motivate our method. The conventional model of image restoration is defined as:

$$\min_x \frac{\lambda}{2}\|y - x * k\|_2^2 + \sum_{l=h,w} \rho(p_l * x), \tag{2}$$

where p_h, p_w are the horizontal and vertical gradient operators, respectively. $\rho(\cdot)$ is the regularization of image gradient of x. With the half-quadratic splitting method, model (2)

Table 1. The architecture of FCNN in one iteration.

name	kernel size	stride	pad	kernel number
conv1	5×5	1	2	64
conv2	3×3	1	1	64
conv3	3×3	1	1	64
conv4	3×3	1	1	64
conv5	3×3	1	1	64
conv6	3×3	1	1	1

can be reformulated as:

$$\min_{x,z} \frac{\lambda}{2}\|y - x * k\|_2^2 + \beta \sum_{l=h,w} \|z_l - p_l * x\|_2^2 + \rho(z_l), \quad (3)$$

where z_l is an auxiliary variable and β is a weight. The half-quadratic optimization with respect to (3) is to alternatively solve:

$$\min_{z} \beta \sum_{l=h,w} \|z_l - p_l * x\|_2^2 + \rho(z_l), \quad (4)$$

and

$$\min_{x} \frac{\lambda}{2}\|y - x * k\|_2^2 + \beta \sum_{l=h,w} \|z_l - p_l * x\|_2^2. \quad (5)$$

We note that (4) is actually a denoising problem while (5) is a deconvolution with respect to x. If the solution for z_l is obtained, the clear image can be efficiently computed by fast Fourier transform (FFT) as:

$$x = \mathcal{F}^{-1}\left(\frac{\gamma\overline{\mathcal{F}(k)}\mathcal{F}(y) + \sum_{l=h,w}\overline{\mathcal{F}(p_l)}\mathcal{F}(z_l)}{\gamma\overline{\mathcal{F}(k)}\mathcal{F}(k) + \sum_{l=h,w}\overline{\mathcal{F}(p_l)}\mathcal{F}(p_l)}\right), \quad (6)$$

where $\mathcal{F}(\cdot)$ and $\mathcal{F}^{-1}(\cdot)$ denote the Fourier transform and its inverse transform, respectively. $\overline{\mathcal{F}(\cdot)}$ is the complex conjugate of Fourier transform, and $\gamma = \frac{\lambda}{2\beta}$ is the hyperparameter in deconvolution.

We note that the main problem is how to define a good image prior for (4). In the following, we propose an effective algorithm based on FCNN to learn an effective image prior for (4).

3.2. Network Architecture

The proposed network architecture for non-blind deconvolution is shown in Figure 1. The input of our network includes a blurry image and the corresponding blur kernel. The proposed network first applies the deconvolution operation on the blurry image via a deconvolution module and then performs convolutions to the vertical and horizontal gradients to generate the results with less noise. The denoised image gradients are treated as image priors to guide the image deconvolution in the next iteration.

Denoising by FCNN. We note that though the output of deconvolution x from (6) is sharp, it usually contains noise and significant ringing artifacts (see Figure 2(k)). To solve this problem, we develop a FCNN and apply it to the vertical and horizontal gradients to remove noise and ringing artifacts. Applying FCNN to the vertical gradients and horizontal gradients usually leads to different network weight parameters. Similar to [36], we transpose the vertical gradients so vertical and horizontal gradients can share the weights in the training process. Table 1 shows the details of the proposed network in one iteration. We add a rectified linear unit (ReLU) after every convolution layer as activation function except the last one. Although we use the same network architecture in different iterations, the weights are different for different iterations.

Deconvolution module. The deconvolution module is used to restore the sharp image. It is defined by (5). In the proposed network, it is applied to the gradient denoising outputs from FCNN to guide image restoration.

3.3. Loss Function for FCNN Training

Since it is very difficult to train the network in an end-to-end manner, we iteratively train the weights of FCNN. That is, we first train the network weights and then fix these weights when performing the deconvolution. After the deconvolution module, we train the network weights in the next iteration. This training procedure is achieved by minimizing the loss function L:

$$L(\bigtriangledown_h x, \bigtriangledown_w x, x_0; \theta) = \frac{1}{N}\sum_{i=1}^{N}(\|f(\bigtriangledown_h x^{(i)}; \theta) - \bigtriangledown_h x_0^{(i)}\|_1 + \|f(\bigtriangledown_w x^{(i)}; \theta) - \bigtriangledown_w x_0^{(i)}\|_1), \quad (7)$$

where $f(\cdot)$ is the denoising mapping learned by FCNN, θ is the FCNN weights, $\bigtriangledown_h x = p_h * x$, $\bigtriangledown_w x = p_w * x$, N is the number of training samples in every batch, $\|\cdot\|_1$ is L_1 norm and x_0 is the ground truth image.

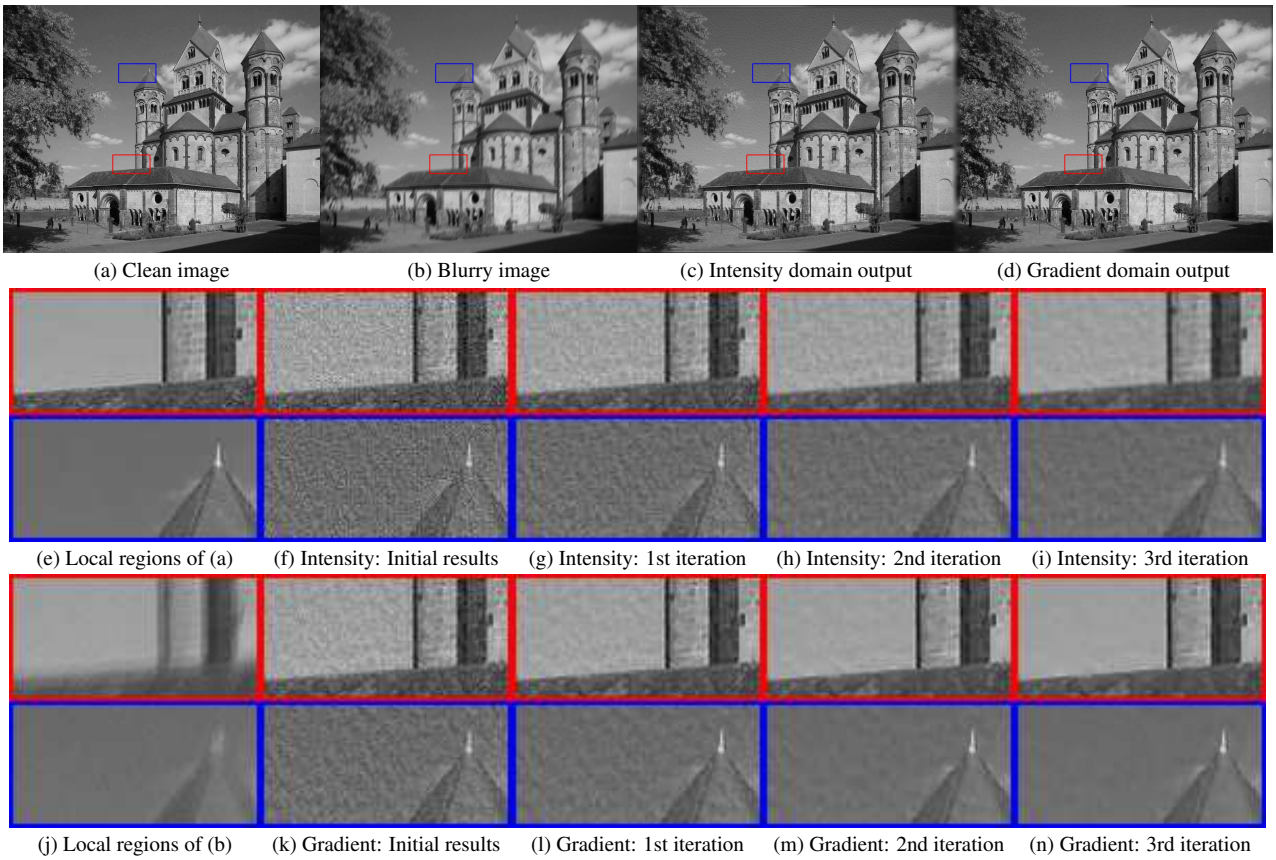
3.4. Hyper-parameters Training

In order to get optimal hyper-parameters γ for the deconvolution module (5), we train them in an end-to-end manner with fixed FCNN weights. The hyper-parameters training process is achieved by minimizing the loss function as:

$$\mathcal{L}^h = \frac{1}{N}\sum_{i}^{N}\|x^{(i)} - x_0^{(i)}\|_1, \quad (8)$$

where x is the output of the final deconvolution module.

As the forward propagation of the deconvolution module is defined by (6), we can obtain the gradient in backward



(a) Clean image (b) Blurry image (c) Intensity domain output (d) Gradient domain output

(e) Local regions of (a) (f) Intensity: Initial results (g) Intensity: 1st iteration (h) Intensity: 2nd iteration (i) Intensity: 3rd iteration

(j) Local regions of (b) (k) Gradient: Initial results (l) Gradient: 1st iteration (m) Gradient: 2nd iteration (n) Gradient: 3rd iteration

Figure 2. Visual results produced after each iteration in different domains. (a) shows the clean image. (b) shows the blurry image. (c) and (d) are results produced from the intensity and gradient domains, respectively. (e) and (f) are the extracted local regions of (a) and (b), respectively. (f) shows the initial deconvolution result when the network is applied on the intensity domain. (g)-(i) show the corresponding refined results through different numbers of iterations. (k) shows the initial deconvolution result when the network is applied on the gradient domain. (l)-(n) show the corresponding refined results through different numbers of iterations. This demonstrates that our network is more effective in reducing noise through several iterations in the gradient domain.

propagation by

$$\Delta_{z_l} = \mathcal{F}\left(\frac{\overline{\mathcal{F}(p_l)}\mathcal{F}^{-1}(\Delta_x)}{\gamma\overline{\mathcal{F}(k)}\mathcal{F}(k) + \sum_{l=h,w}\overline{\mathcal{F}(p_l)}\mathcal{F}(p_l)}\right), \quad (9)$$

where $\Delta_x = \frac{x^{(i)} - x_0^{(i)}}{|x^{(i)} - x_0^{(i)}|}$.

The gradient that is used to train the hyper-parameters γ can be written as:

$$\Delta_\gamma = \left(\frac{\mathbf{DH} - \mathbf{EG}}{(\gamma\mathbf{G} + \mathbf{H})^2}\right)^\top \mathbf{L}_x^h, \quad (10)$$

where $\mathbf{D}$, $\mathbf{H}$, $\mathbf{E}$, $\mathbf{G}$, and $\mathbf{L}_x^h$ denote the vector forms of D, H, E, G and $\mathcal{L}_x^h$, respectively, in which $D = \overline{\mathcal{F}(k)}\mathcal{F}(y)$, $E = \sum_{l=h,w}\overline{\mathcal{F}(p_l)}\mathcal{F}(z_l)$, $G = \overline{\mathcal{F}(k)}\mathcal{F}(k)$, $H = \sum_{l=h,w}\overline{\mathcal{F}(p_l)}\mathcal{F}(p_l)$, and $L_x^h = \mathcal{F}^{-1}(\Delta_x)$. The detailed derivations of (9) and (10) are included in the supplemental material.

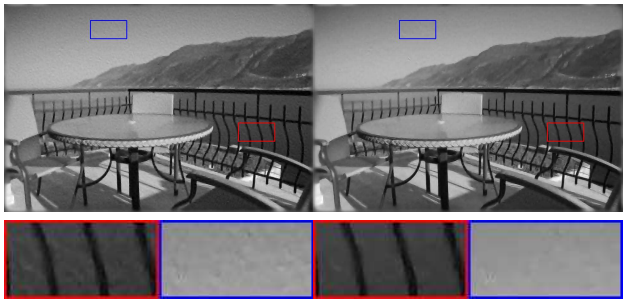
4. Analysis and Discussion

In this section, we analyze the effect of the iterative-wise FCNN, show why we use the gradient domain, and validate the proposed loss function used in the proposed network.

4.1. Effect of the Iterative-wise FCNN

In the proposed method, we iteratively solve the deconvolution and denoising part. That is, the network parameters of FCNN are trained at each iteration. With this manner, the high-quality results can be obtained.

Figure 3 shows an example which demonstrates the effectiveness of the iterative-wise FCNN. As shown in Figure 3(a), the result generated by one-iteration network contains some artifacts and has a lower PSNR value. In contrast, these artifacts are reduced by the iterative FCNN which accordingly lead to a much clearer image with higher PSNR value (Figure 3(b)). Noted that one-iteration network is different from the first iteration stage of the proposed three-iteration network such as Figure 2(l). We optimize the



(a) One-iteration network PSNR: 29.26 dB

(b) Three-iteration network PSNR: 29.75 dB

Figure 3. The effectiveness of the iterative-wise FCNN. Using only one iteration does not remove the noise. See Section 4.1 for details.

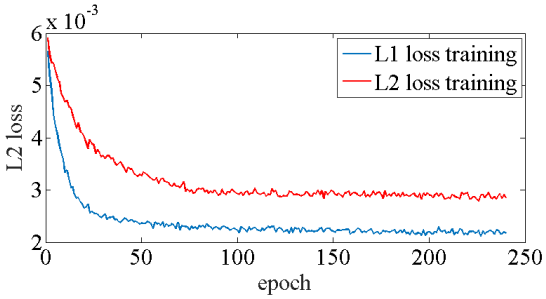


Figure 4. The 1st iteration FCNN denoising gradient training loss with 1% noise. The converged L_2 norm training loss is higher than L_1 norm.

FCNN weights and deconvolution hyper-parameters for the network with only one iteration. More quantitative results are presented in Section 5.3 with different noise levels.

4.2. Gradient Domain versus Intensity Domain

The denoising part is mainly used to remove noise and ringing artifacts while keeping textures. We note that the image gradient is able to model the details and structures of images. Thus, we train the network in image gradient domain instead of intensity domain. We train two three-iteration networks to generate the restored results based on intensity domain and gradient domain respectively. As shown in Figure 2, the results reconstructed from intensity domain (the second row) contain several noise and artifacts relative to that from gradient domain (the third row).

4.3. Effect of the Loss Function

Most of existing CNN based low-level vision methods use the L_2 norm based reconstruction error as the loss function *e.g.* [6]. However, the L_2 norm is not robust to outliers and usually leads to results contain noise and ringing artifacts [38]. To overcome the limitations of L_2 norm based reconstruction error, we use an L_1 norm based reconstruction error as the loss function, *i.e.*, (7).

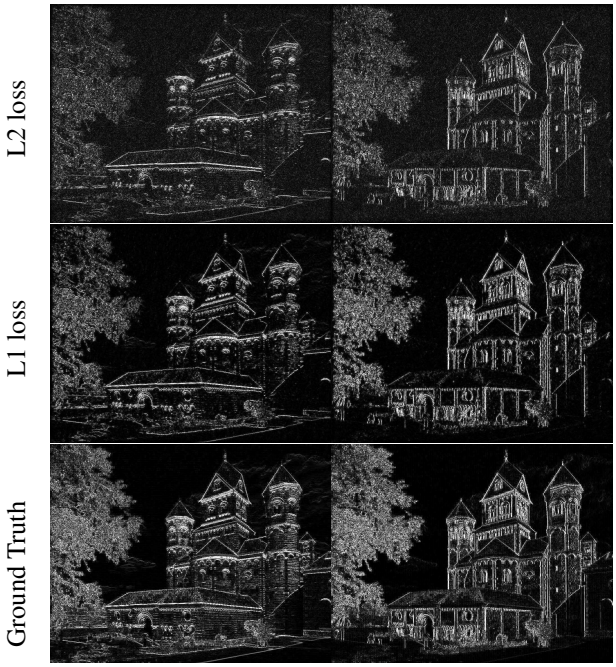


Figure 5. Visual comparison of gradient generation under different loss functions. The input blurry image is with 1% noise. The vertical and horizontal gradient trained using L_2 loss and L_1 loss are shown in the first and second rows, respectively. The ground truth gradient is shown in the last row. The gradient noise can be effectively reduced using L_1 loss.

To validate the effect of the L_1 norm loss function, we train the proposed network with the L_2 norm based reconstruction error and the L_1 norm based reconstruction error using the same settings for the first iteration. As shown in Figure 4, the method with the L_1 norm based reconstruction error converges better than that of the L_2 norm based reconstruction error. Figure 5 shows that using L_1 norm based reconstruction error is able to remove noise and artifacts compared to that of L_2 norm based reconstruction error.

5. Experimental Results

We evaluate the proposed algorithm against the state-of-the-art methods using benchmark datasets for non-blind deconvolution. The MATLAB code is available at https://sites.google.com/site/zhjw1988/.

5.1. Training

Parameter settings. To train the network, we optimize the hyper-parameters and the weights of FCNN iteratively. Specifically, the weights of FCNN are trained iteration by iteration with fixed hyper-parameters and then hyper-parameters are trained in an end-to-end manner with fixed

Figure 6. Examples of randomly generated kernels for training.

weights of FCNN . We implement the proposed algorithm using the MatConvNet [31] toolbox. We use Xavier initialization for the FCNN weights of each layer. The initial values in the hyper-parameters training stage are randomly initialized. (But keeping the later iteration has smaller hyper-parameter than the former one.) Stochastic gradient descent (SGD) is used to train the network. The learning rate in the training of FCNN is 0.01. For the learning rate in the hyper-parameter training stage, we set it to be 10 in the last deconvolution module and 10,000 in other modules. The momentum value for both FCNN and hyper-parameters training is set to be 0.95. Since hyper-parameters are easily stuck into local minimal. We train the network with several hyper-parameter initializations and select the best one.

Training dataset. In order to generate enough blurred images for training, we use BSD500 dataset [1]. and randomly crop image patches with a size of 256×256 pixels as the clear images. The blurred kernels are generated according to [4], whose size ranges from 11 to 31 pixels. We generate blurred kernels according to [4], where the size of blur kernels ranges from 11 to 31 pixels. Some examples of our randomly generated kernels are shown in Figure 6. After obtaining these generated blur kernels, we convolve the clear image patches with the blur kernels and Gaussian noise to obtain the blurry image patches. We also train three networks with 1%, 3% and 5% of noise.

Test dataset. For the test dataset, we use the 80 ground truth clear images from the dataset by Sun *et al*. [29] and eight blur kernels from the dataset by Levin *et al*. [15]. Thus, we have 640 blurred images in total. We evaluate all the methods on the blurred images with different Gaussian noise level which includes 1%, 3% and 5%. In addition to use the ground truth kernels from the dataset by Levin *et al*. [15], we also use the estimated blur kernels from the state-of-the-art blind deblurring method [18] to examine the effectiveness of the proposed method.

5.2. Convergence Property

We quantitatively evaluate the convergence properties of our method and empirically find that our method converges well after three iterations as shown in Table 2. More iterations do not generate better restoration results.

5.3. Comparisons with the State-of-the-Arts

We compare the proposed iterative FCNN with other non-blind deblurring algorithms including HL [14], EPLL [39], MLP [28], and CSF [25]. For the proposed method, we also use the proposed algorithm with one-iteration and three-iteration network for comparison. For fairness, we use the online available implementation of these methods and tuned the parameters to generate the best possible results.

We first quantitatively evaluate the proposed method on the dataset with 1% Gaussian noise using PSNR and SSIM

Table 2. Average sum of squared differences (SSD) error of FCNN output from different iterations with different noise levels. The output error does not change significantly after two iterations.

iteration	1st	2nd	3rd
1%	0.0323	0.0313	0.0312
3%	0.0436	0.0422	0.0419
5%	0.0477	0.0463	0.0459

Table 3. Average PSNR and SSIM for 1% noise.

blur kernel	ground truth	Pan [18]
HL [14]	31.57/0.87	29.94/0.84
EPLL [39]	33.00/0.89	30.61/0.87
MLP [28]	31.82/0.86	28.76/0.80
CSF [25]	31.93/0.87	30.22/0.86
1-iteration	32.50/0.89	30.38/0.86
3-iteration	32.82/0.90	30.39/0.87

Table 4. Average PSNR and SSIM for 3% noise.

blur kernel	ground truth	Pan [18]
HL [14]	27.42/0.73	26.91/0.72
EPLL [39]	28.71/0.78	27.75/0.77
MLP [28]	26.26/0.60	25.04/0.57
CSF [25]	28.43/0.78	27.11/0.74
1-iteration	28.71/0.77	27.34/0.75
3-iteration	29.05/0.79	27.74/0.77

Table 5. Average PSNR and SSIM for 5% noise.

blur kernel	ground truth	Pan [18]
HL [14]	25.85/0.67	25.48/0.66
EPLL [39]	27.00/0.71	26.24/0.71
MLP [28]	24.62/0.51	22.32/0.45
CSF [25]	26.92/0.67	24.86/0.65
1-iteration	27.25/0.72	25.49/0.69
3-iteration	27.46/0.74	26.33/0.72

Table 6. Average PSNR and SSIM with ground truth kernels and different noise levels.

noise level	1%	3%	5%
IDDBM3D [5]	32.88/0.89	29.00/0.79	27.43/0.73
NCSR [7]	32.78/0.89	27.69/0.66	24.79/0.49
3-iteration	32.82/0.90	29.05/0.79	27.46/0.74

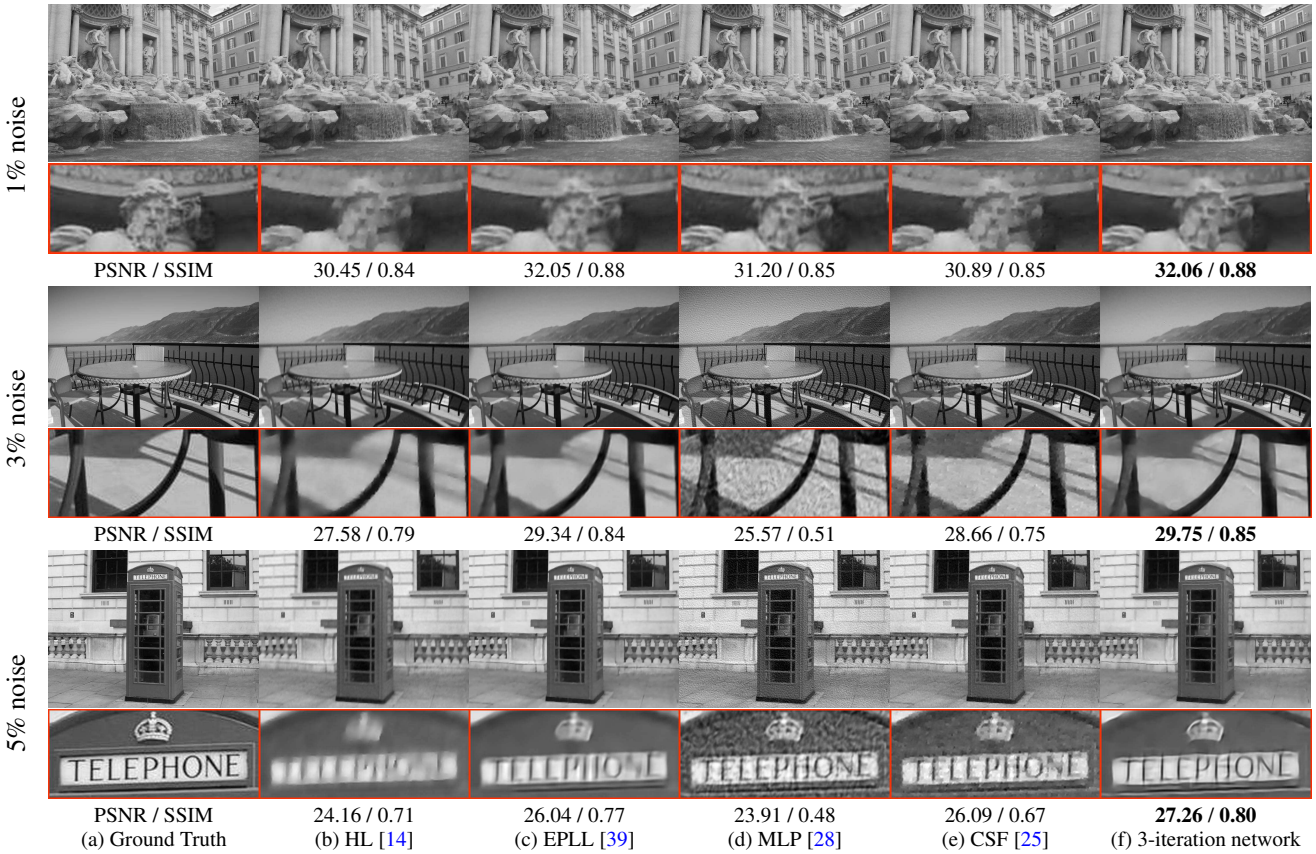


Figure 7. Visual evaluation under different input noise levels. The proposed method performs favorably compared with existing non-blind deblurring methods.

Table 7. Average time cost (seconds) with different image sizes for three-iteration network. HL and EPLL run on a Intel Core i7 CPU and MLP, CSF and our method run on a Nvidia K40 GPU.

image size	HL [14]	EPLL [39]	MLP [28]	CSF [25]	ours
512×400	0.31	209.58	0.80	0.08	0.02
1024×800	0.71	953.52	2.98	0.09	0.03
1536×1200	2.11	N/A[a]	6.73	0.33	0.06

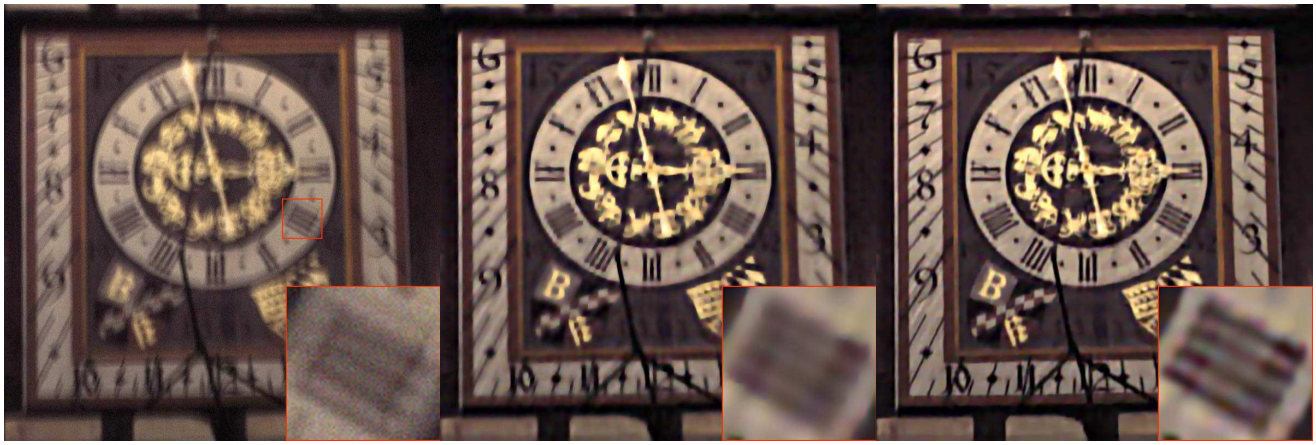
[a]Our computer does not have enough memory to deconvolute a 1536×1200 image by EPLL.

as the metrics. As shown in Table 3, our method outperforms HL [14], MLP [28] and CSF [25] in terms of PSNR and SSIM metrics. Although EPLL method performs slightly better than the proposed method, this method is not efficient as it needs to solve complex optimization problems. Furthermore, this method usually smooths details as shown in Figure 7(c), while the proposed method generates the results with much clearer textures (Figure 7(f)). We further note that the PSNR and SSIM values of the proposed iterative FCNN are higher than those of the proposed method with only one iteration, which demonstrates the effectiveness of the iterative FCNN method. In addition, we use the estimated blur kernels from Pan et al. [18] to evaluate the proposed method. The PSNR and SSIM values in Table 3 demonstrate that the proposed method still performs well and can be applied to method Pan et al. [18] to improve the performance of restored results.

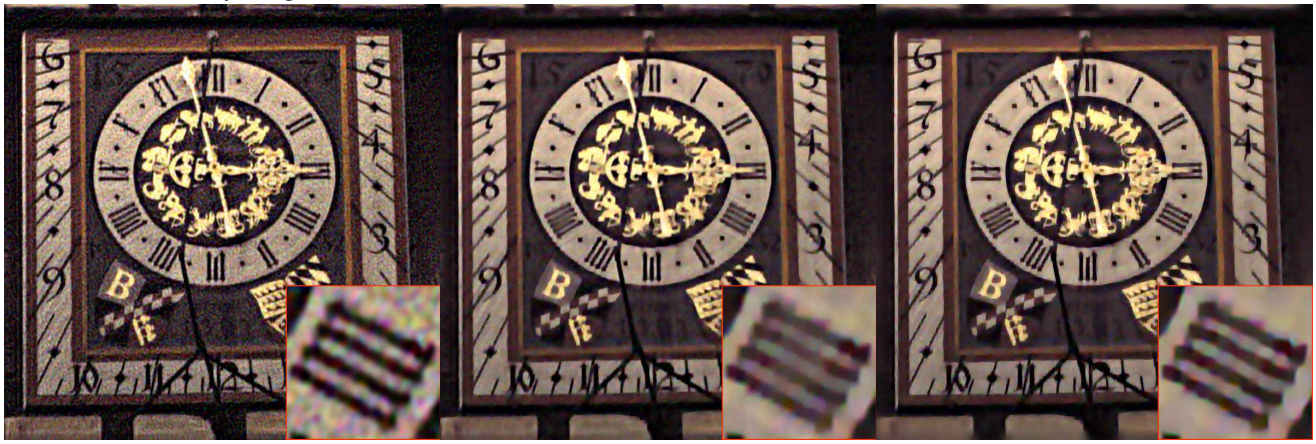
We further evaluate our method on the images with 3% and 5% Gaussian noise. Tables 4 and 5 show the results by different methods. Our method achieves better performance compared with HL [14], MLP [28] and CSF [25] when the noise level is high. In addition to PSNR and SSIM, our method also generates much clearer images with fine textures as shown in Figure 7.

We also compare the proposed method with IDDBM3D [5] and NCSR [7] under different noise levels in Table 6. Since IDDBM3D and NCSR take about 20 and 35 minutes respectively to deconvolute a 800×1024 image, we only compare the proposed method with them with ground truth kernels. It shows that the proposed algorithm achieves comparable results to the IDDBM3D and NCSR.

Runtime. The proposed method performs favorably against



(a) blurry image (b) HL [14] (c) EPLL [39]



(d) MLP [28] (e) CSF [25] (f) 3-iteration network

Figure 8. Visual evaluation on the image from [13]. We have added 3% Gaussian noise. The proposed method performs favorably compared with existing non-blind deblurring methods.

other state-of-the-art methods in terms of runtime. Table 7 summarizes the average runtime of representative methods with different image resolutions. HL and EPLL are run on an Intel Core i7 CPU and MLP, CSF and the proposed method are run on a Nvidia K40 GPU.

5.4. Results on Real Blurry Images

We also test our three-iteration network for one real blurry image from [13]. We add 3% Gaussian noise to the original blurred image and use the network trained with this noise level for this experiment. We use [18] to estimate kernel of the blurry image. Figure 8 shows that CSF cannot remove all the noise especially in flat regions and the result of HL still contains blur residual. In contrast, our three-iteration network achieves comparable performance compared to EPLL.

6. Conclusion

We propose an efficient non-blind deconvolution algorithm based on a fully convolutional neural network (FCNN). The proposed method involves deconvolution part and denoising part, where the denoising part is achieved by a FCNN. The learned features from FCNN is able to help the deconvolution. To remove noise and ringing artifacts, we develop an iterative-wise FCNN, which is able to preserve image details. Furthermore, we propose a hyper-parameters learning algorithm to improve the performance of image restoration. The proposed method performs favorably against state-of-the-art methods on both synthetic and real-world images in terms of both quality and speed.

Acknowledgements. This work is supported in part by the SRG grant from City University of Hong Kong (No. 7004416), the National Natural Science Foundation of China (No. 61572099 and 61320106008), the NSF Career Grant 1149783 and gifts from Adobe and Nvidia.

References

[1] P. Arbelaez, M. Maire, C. Fowlkes, and J. Malik. Contour detection and hierarchical image segmentation. *TPAMI*, 33(5):898–916, 2011. 6

[2] A. Buades, B. Coll, and J. Morel. A non-local algorithm for image denoising. In *CVPR*, pages 60–65, 2005. 1

[3] H. C. Burger, C. J. Schuler, and S. Harmeling. Image denoising: Can plain neural networks compete with bm3d? In *CVPR*, 2012. 2

[4] A. Chakrabarti. A neural approach to blind motion deblurring. In *ECCV*, 2016. 6

[5] A. Danielyan, V. Katkovnik, and K. Egiazarian. Bm3d frames and variational image deblurring. *TIP*, 21(4):1715–1728, 2012. 6, 7

[6] C. Dong, C. C. Loy, K. He, and X. Tang. Learning a deep convolutional network for image super-resolution. In *ECCV*, 2014. 2, 5

[7] W. Dong, L. Zhang, G. Shi, and X. Li. Nonlocally centralized sparse representation for image restoration. *TIP*, 22(4):1620–1630, 2013. 6, 7

[8] D. Eigen, D. Krishnan, and R. Fergus. Restoring an image taken through a window covered with dirt or rain. In *ICCV*, 2013. 2

[9] R. Fergus, B. Singh, A. Hertzmann, S. T. Roweis, and W. T. Freeman. Removing camera shake from a single photograph. *ACM TOG (Proc. SIGGRAPH)*, 25(3):787–794, 2006. 2

[10] V. Jain and S. Seung. Natural image denoising with convolutional networks. In *NIPS*, 2009. 2

[11] J. Kim, J. Lee, and K. Lee. Accurate image super-resolution using very deep convolutional networks. In *CVPR*, 2016. 2

[12] J. Kim, J. Lee, and K. Lee. Deeply-recursive convolutional network for image super-resolution. In *CVPR*, 2016. 2

[13] R. Köhler, M. Hirsch, B. Mohler, B. Schölkopf, and S. Harmeling. Recording and playback of camera shake: Benchmarking blind deconvolution with a real-world database. In *ECCV*, 2012. 8

[14] D. Krishnan and R. Fergus. Fast image deconvolution using hyper-laplacian priors. In *NIPS*, 2009. 1, 2, 6, 7, 8

[15] A. Levin, Y. Weiss, F. Durand, and W. T. Freeman. Understanding and evaluating blind deconvolution algorithms. In *CVPR*, 2009. 1, 2, 6

[16] Y. Li, J.-B. Huang, N. Ahuja, and M.-H. Yang. Deep joint image filtering. In *ECCV*, 2016. 2

[17] S. Liu, J. Pan, and M.-H. Yang. Learning recursive filters for low-level vision via a hybrid neural network. In *ECCV*, 2016. 2

[18] J. Pan, Z. Lin, Z. Su, and M.-H. Yang. Robust kernel estimation with outliers handling for image deblurring. In *CVPR*, 2016. 6, 7, 8

[19] W. Ren, S. Liu, H. Zhang, J. Pan, X. Cao, and M.-H. Yang. Single image dehazing via multi-scale convolutional neural networks. In *ECCV*, 2016. 2

[20] W. Richardson. Bayesian-based iterative method of image restoration. *JOSA*, 62(1):55–59, 1972. 1

[21] G. Riegler, D. Ferstl, M. Rüther, and H. Bischof. A deep primal-dual network for guided depth super-resolution. In *BMVC*, 2016. 2

[22] G. Riegler, M. Rüther, and H. Bischof. Atgv-net: Accurate depth super-resolution. In *ECCV*, 2016. 2

[23] S. Roth and M. Black. Fields of experts: A framework for learning image priors. In *CVPR*, pages 860–867, 2005. 1, 2

[24] U. Schmidt, J. Jancsary, S. Nowozin, S. Roth, and C. Rother. Cascades of regression tree fields for image restoration. *TPAMI*, 38(4):677–689, 2016. 1

[25] U. Schmidt and S. Roth. Shrinkage fields for effective image restoration. In *CVPR*, 2014. 1, 2, 6, 7, 8

[26] U. Schmidt, C. Rother, S. Nowozin, J. Jancsary, and S. Roth. Discriminative non-blind deblurring. In *CVPR*, 2013. 1, 2

[27] U. Schmidt, K. Schelten, and S. Roth. Bayesian deblurring with integrated noise estimation. In *CVPR*, pages 2625–2632, 2011. 1

[28] C. J. Schuler, H. Christopher Burger, S. Harmeling, and B. Scholkopf. A machine learning approach for non-blind image deconvolution. In *CVPR*, 2013. 1, 2, 6, 7, 8

[29] L. Sun, S. Cho, J. Wang, and J. Hays. Edge-based blur kernel estimation using patch priors. In *ICCP*, 2013. 6

[30] L. Sun, S. Cho, J. Wang, and J. Hays. Good image priors for non-blind deconvolution - generic vs. specific. In *ECCV*, pages 231–246, 2014. 1, 2

[31] A. Vedaldi and K. Lenc. Matconvnet – convolutional neural networks for matlab. In *ACM MM*, 2015. 6

[32] Z. Wang, D. Liu, J. Yang, W. Han, and T. Huang. Deep networks for image super-resolution with sparse prior. In *ICCV*, 2015. 2

[33] N. Wiener. *Extrapolation, interpolation, and smoothing of stationary time series*, volume 2. MIT Press, 1949. 1

[34] J. Xie, L. Xu, and E. Chen. Image denoising and inpainting with deep neural networks. In *NIPS*, 2012. 2

[35] L. Xu, J. S. Ren, C. Liu, and J. Jia. Deep convolutional neural network for image deconvolution. In *NIPS*, 2014. 1, 2

[36] L. Xu, J. S. Ren, Q. Yan, R. Liao, and J. Jia. Deep edge-aware filters. In *ICML*, 2015. 2, 3

[37] X. Yu and F. Porikli. Ultra-resolving face images by discriminative generative networks. In *ECCV*, 2016. 2

[38] H. Zhao, O. Gallo, I. Frosio, and J. Kautz. Loss functions for neural networks for image processing. *TCI*, 3(1):47–57, 2017. 5

[39] D. Zoran and Y. Weiss. From learning models of natural image patches to whole image restoration. In *ICCV*, 2011. 1, 2, 6, 7, 8