Neural Blind Deconvolution Using Deep Priors
Supplementary Material

Dongwei Ren[1], Kai Zhang[2], Qilong Wang[1], Qinghua Hu[1], and Wangmeng Zuo[2]

[1]College of Intelligence and Computing, Tianjin University, Tianjin, China
[2]School of Computer Science and Technology, Harbin Institute of Technology, Harbin, China

This supplementary file includes details of network architecture, comparison on benchmark datasets and more results on real blurry images.

1. Network Architecture

1.1. Architecture of $\mathcal{G}_k$

Due to the sparsity of blur kernel $\mathbf{k}$, we only use simple fully connected network to implement $\mathcal{G}_k$. The *SoftMax* is used to meet the normalization constraint of blur kernel. The 1D output of $\mathcal{G}_k$ is finally reshaped to 2D blur kernel.

Table s1. The architecture of $\mathcal{G}_k$. Fully connected layer is with the form Linear(input channel, output channel).

Input: kernel size $m_k \times n_k$, $\mathbf{z}_k$ (200) from the uniform distribution with seed 0.	
Output: blur kernel $\mathbf{k}$ with size $m_k \times n_k$.	
Hidden layer	Linear(200, 1000); *ReLU*
Output layer	Linear(1000, $m_k \times n_k$); *SoftMax*

1.2. Architecture of $\mathcal{G}_x$

As for $\mathcal{G}_x$, we employ the encoder-decoder network with skip connections [5]. As shown in Fig. s1, the i-th unit encoder-decoder architecture is first demonstrated. Taking e_i as an example, we use the form $e_i(n_f, k, p)$ to represent that the convolutions in e_i have n_f filters with size $k \times k$ and $p \times p$ padding. We note that the filter size in the last convolution of d_i is fixed as 1×1. The slope of *LeakyReLU* is 0.2, downsampling is implemented using $stride = 2$, and upsampling is implemented using 2x *bilinear* interpolation. The generative network $\mathcal{G}_x$ have five units in Fig. s1, and its parameter settings are detailed presented in Table s2. In our current implementation, $\mathcal{G}_x$ is used to only process the y channel given a color image in ycbcr space.

Table s2. The architecture of $\mathcal{G}_x$, which consists of five e_i, d_i and s_i in Fig. s1. Convolution is with the form Conv.(input channel, output channel, kernel size, padding size)

Input: $\mathbf{z}_x$ ($8 \times m_x \times n_x$) from the uniform distribution with seed 0.	
Output: latent image $\mathbf{x}$ ($1 \times m_x \times n_x$).	
Encoder unit 1	$e_1(128, 3, 1)$, $s_1(16, 3, 1)$
Encoder unit 2	$e_2(128, 3, 1)$, $s_2(16, 3, 1)$
Encoder unit 3	$e_3(128, 3, 1)$, $s_3(16, 3, 1)$
Encoder unit 4	$e_4(128, 3, 1)$, $s_4(16, 3, 1)$
Encoder unit 5	$e_5(128, 3, 1)$, $s_5(16, 3, 1)$
Decoder unit 5	$d_5(128, 3, 1)$
Decoder unit 4	$d_4(128, 3, 1)$
Decoder unit 3	$d_3(128, 3, 1)$
Decoder unit 2	$d_2(128, 3, 1)$
Decoder unit 1	$d_1(128, 3, 1)$
Output layer	Conv.(128,1,1,0); *Sigmoid*

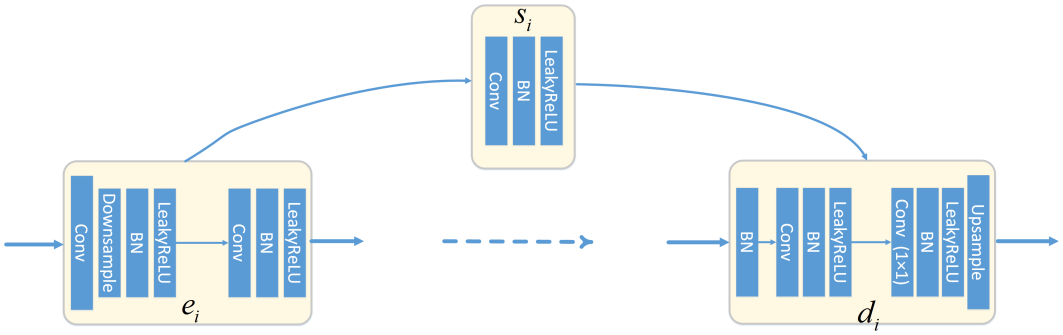
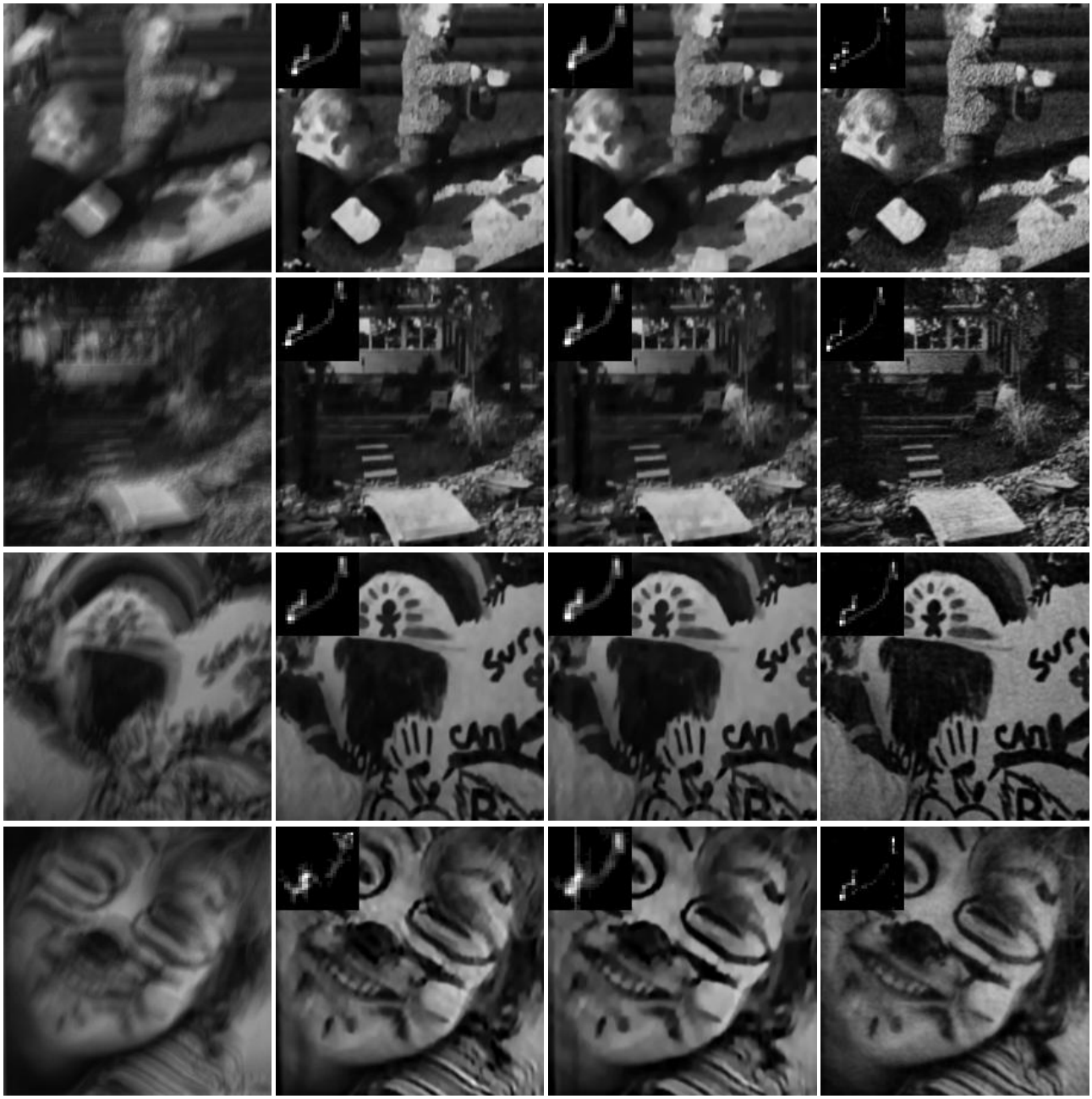


Figure s1. One unit in encoder-decoder with skip connection, where e_i is the i-th unit in encoder, d_i is the i-th unit in decoder and s_i is the skip connection between e_i and d_i. Taking e_i as an example, we use the form $e_i(n_f, k, p)$ to represent that the convolutions in e_i have n_f filters with size $k \times k$ and $p \times p$ padding. We note that the filter size in the last convolution of d_i is fixed as 1×1. The slope of *LeakyReLU* is 0.2, downsampling is implemented using $stride = 2$, and upsampling operation is implemented as 2x *bilinear* interpolation.

2. Comparison on Benchmarks

2.1. Results on dataset of Levin *et al*. [2]

In Fig. s2, we compare the deblurring results on #4 kernel, which is the most difficult to handle. The deblurring results by our SelfDeblur are with finer textures, while the results by the other methods are often smoothing.



Blurry images | Sun *et al*. [4] | Pan-DCP [3] | SelfDeblur

Figure s2. Comparison on Levin *et al*.'s dataset.

2.2. Results on dataset of Lai *et al*. [1]

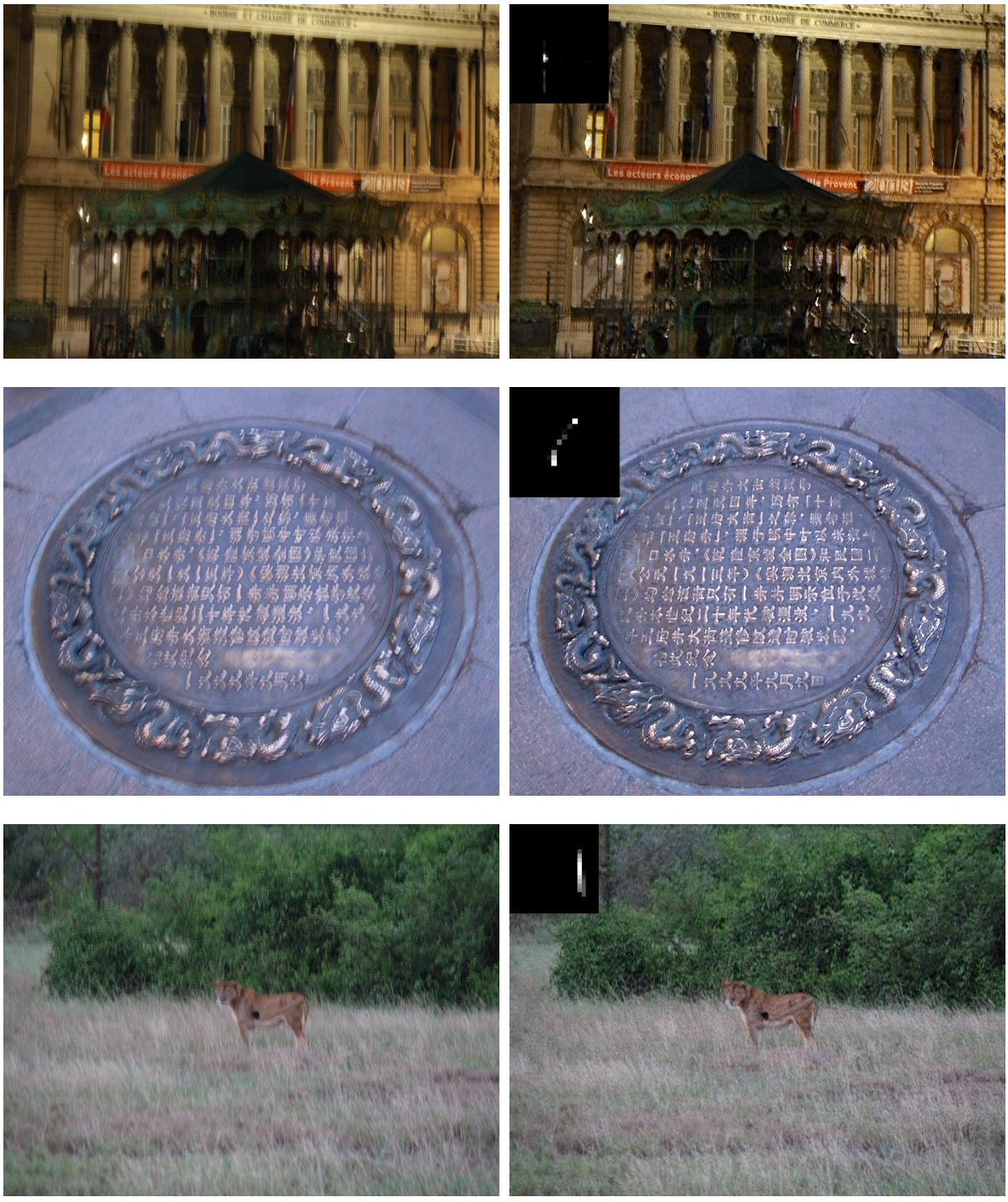
In Fig. s3, we demonstrate comparison results from 5 categories, *i.e*., *Manmade*, *Natural*, *People*, *Saturated* and *Text*. The images from this dataset are usually with high resolution, so please zoom in the deblurring results to compare texture details.

Blurry images | Xu&Jia [6] | Pan-DCP [3] | SelfDeblur

Figure s3. Comparison on Lai *et al*.'s dataset. Please zoom in to see texture details.

3. More Results on Real Blurry Images

The left is blurry images, and the right presents the estimated blur kernels and deblurring results by our SelfDeblur.



References

[1] W.-S. Lai, J.-B. Huang, Z. Hu, N. Ahuja, and M.-H. Yang. A comparative study for single image blind deblurring. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1701–1709, 2016. 4

[2] A. Levin, Y. Weiss, F. Durand, and W. T. Freeman. Understanding and evaluating blind deconvolution algorithms. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1964–1971. IEEE, 2009. 3

[3] J. Pan, D. Sun, H. Pfister, and M.-H. Yang. Deblurring images via dark channel prior. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(10):2315–2328, 2018. 3, 4

[4] L. Sun, S. Cho, J. Wang, and J. Hays. Edge-based blur kernel estimation using patch priors. In *IEEE International Conference on Computational Photography*, pages 1–8. IEEE, 2013. 3

[5] D. Ulyanov, A. Vedaldi, and V. Lempitsky. Deep image prior. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 9446–9454, 2018. 1

[6] L. Xu and J. Jia. Two-phase kernel estimation for robust motion deblurring. In *European Conference on Computer Vision*, pages 157–170. Springer, 2010. 4