Non-Blind Deblurring for Fluorescence: A Deformable Latent Space Approach with Kernel Parameterization

Ziqiao Guan
Stony Brook University
ziguan@cs.stonybrook.edu

Esther H. R. Tsai
Brookhaven National Laboratory
etsai@bnl.gov

Xiaojing Huang
Brookhaven National Laboratory
xjhuang@bnl.gov

Kevin G. Yager
Brookhaven National Laboratory
kyager@bnl.gov

Hong Qin
Stony Brook University
qin@cs.stonybrook.edu

Abstract

Non-blind deblurring (NBD) is a modeling method of the image deblurring problem in computer vision, where the blurring kernel is known or can be externally estimated. In this paper, we attempt to solve a parametric NBD problem, inspired by the simultaneous acquisition of ptychography and fluorescent imaging (FI). Ptychography is an imaging method that favors larger probes, i.e. *convolutional kernels, while FI relies on a small probe for high resolution. Also, the kernel can be solved during ptychographic reconstruction. With Ptycho-FI using the same larger kernel, we can perform NBD on the blurred fluorescent images to achieve high-resolution FI, and thus speed up the experiments. To this end, we design a deep latent space deformation network that is directly parameterized by the kernel. The network consists of three components: encoder, deformer, and decoder, where the deformer is specifically meant to rectify the latent space representations of blurred images to a standard latent space, regardless of the kernel. The deformation network is trained with a two-stage training scheme. We conduct extensive experiments to confirm that our parametric model can adapt to drastically different blurring kernels and perform robust deblurring.*

1. Introduction and Motivation

Image deblurring is a classic problem in photography and imaging science. One of the key elements that defines different approaches is blind deblurring (BD) *vs*. non-blind deblurring (NBD). In a convolutional model, a clean image (latent image) is corrupted by a convolutional blurring kernel and some noise, which is written as

$$B = I * K + N, \tag{1}$$

where B is the blurred image, I is the clean image, K is the kernel, and N is noise. BD refers to recovering both I and K, given the kernel is unknown; and NBD means recovering I with some estimate of K provided. While NBD relieves the ill-posed nature of deblurring, there is yet no direct inverse due to the presence of noise. Also, having more diverse kernels opens up more dramatically different blurring effects, presenting new challenges to improve NBD.

We study a multi-modality imaging experiment – simultaneous acquisition of ptychography and fluorescent imaging (FI), or Ptycho-FI for short – that can use a robust NBD method. Here is a brief description of the setup:

- *Ptychography* captures spatial maps of a sample. It involves scanning the sample with a local probe, *i.e*. convolutional kernel (as in Eqn. 1), offering high-resolution imaging in X-ray, electron, and optical regimes [6, 20]. The resolution of ptychography is not limited by the probe size; the probe can actually be solved during the reconstruction.
- *Fluorescence* offers complementary elemental maps, *e.g*. the distribution of iron in catalyst particles used in the petroleum industry [8]. The FI resolution is limited by the probe size, and a smaller probe has to be used.
- Normally, Ptycho-FI is bottlenecked by the small probe in FI. If we use a larger probe, solve the probe from ptychography, and perform high-quality NBD on FI, this can reduce the acquisition time by a factor of 10^4, and as a result, increase the sample throughput as well as enhance the temporal imaging resolution.

A robust NBD is essential for Ptycho-FI reconstruction, because changing and measuring the kernel K is indeed commonly done in its experiment imaging setup. Generative methods that do not explicitly consider K perform poorly, as we later show in Fig. 2. Unlike existing NBD

methods that resort to regularization [26] or approximation [30] to handle kernel uncertainty, we let the generative model be directly parameterized by K, *i.e*., we will make it real *parametric* NBD.

In this paper, we present a deformable latent space method to perform parametric NBD. The motivation of deblurring in the latent space can be summarized as "shared latent space" – a clean image, and any corrupted version of it, can be embedded in the same low-dimensional latent space. A change in kernel K results in a shift in the latent space. For NBD, we propose to deform the latent space to correct it. The deblurring network can be trained in two stages. First, we establish a standard latent space with one fixed blurring kernel; second, we learn to deform the latent features to the standard latent space, parameterized by the actual blurring kernel. In other words, the deformer parameterizes the latent space with K. Our contributions can be summarized as follows:

- We present a deformable latent space method to perform parametric non-blind deblurring and improve the deblurring quality (Figures 6, 8);
- By operating in the latent space, our NBD method is efficient in terms of time and cost (Table 2);
- We illustrate a parametric non-blind deblurring problem where the varying kernel K takes center stage, and describe the general formulation of shaping a generative method that is flexible, interpretable, and determined by the actual parameters of the imaging process;
- We show the possibility of high-throughput multimodality imaging for advancing material science with the help of NBD.

2. Related Work

Non-Blind Deblurring. Classic NBD methods such as Wiener deconvolution [33] and Richardson-Lucy deconvolution [19] are prone to degradation because of noise. To combat noise, researchers have proposed global, model-driven style priors such as Laplacian, hyper-Laplacian [11], TV regularization [21] and extreme channels prior [35], as well as patch-based, learning-based priors such as BM3D [2], EPLL [41], and pyramid priors [28]. Some learning approaches modify the iteration for faster descent based on learning, such as discriminative NBD [25], shrinkage fields [24], and learned data terms [4]. Generative CNNs are widely used to act as priors and computation blocks in many deblurring methods [34, 38, 37, 12, 30, 18]. Aside from the iterative scheme, there are other methods that address the noise [4], saturated pixels [1], kernel inaccuracies [9, 17], *etc*.. However, not many works prioritize adapting the deblurring model to the changing K.

Latent Space Representation. Latent space is discussed in many works such as denoising autoencoder (DAE) [31] and word2vec [13] for modeling low-dimensional data representations. In image restoration, latent space learned from incomplete data is used in denoising, super-resolution [3] and inpainting [16]. Latent space is also effective for modeling multi-modal data, *e.g*. via multimodal encoder-decoder [15], and canonical correlation analysis (CCA) [7]. We intend to extend latent space models with the parameterization of K.

Deconvolution and Imagery in Science. Extensive developments have been done over past decades to enhance spatial resolution through deconvolution in optical, electron microscopy, and X-ray microscopy [27, 23, 32, 5]. The ability in achieving enhanced imaging resolution often resides in the accurate determination of the kernel for the specific imaging setup and modelling or removing noise in the collected data. In laboratory multi-modality experiments, ptychography allows the accurate determination of the imaging probe, *i.e*. the kernel, and thus presenting the NBD method suitable and advantageous to produce high-resolution fluorescence images from the simultaneously acquired fluorescence data.

3. Deformable Latent Space

In this section, we introduce the formulation of latent space deformation for parametric non-blind deblurring. The network consists of three parts: encoder, deformer and decoder, shown in Fig. 1. Please refer to the supplementary material for the complete technical details of the network.

3.1. Notations

From Ptycho-FI, we obtain low-resolution fluorescent images along with the corresponding blurring kernels measured from ptychography, and then perform NBD. We have a set of clean images $\{\mathbf{I}_i\}$ and a set of blurring kernels $\{k^j\}$. From the images and kernels generate a set of blurred images $\{\mathbf{B}_i^j\}$, where $\mathbf{B}_i^j$ is from image $\mathbf{I}_i$ and kernel k^j (Eqn. 1), and we try to recover $\mathbf{I}_i$. We use subscripts to represent image indices, and superscripts to represent kernel indices.

In Ptycho-FI, not only the kernel k^j is measurable, but we also consider it fully customizable with a set of parameters θ^j, such as defocal distance and pixel density. Thus, k^j can be modeled as $k(\theta^j)$, where θ^j is experiment specific. The deblurring problem is: Recover $\mathbf{I}_i$, given $\mathbf{B}_i^j$ and kernel parameter θ^j.

3.2. Latent Space Shift and Deformation

The notion of latent space is illustrated with a denoising autoencoder (DAE). A DAE consists of downsizing and upsizing layers, which forms a "bottleneck" low-dimensional layer in the middle. The bottleneck forces a compact representation that extracts structure and resists noise. In other

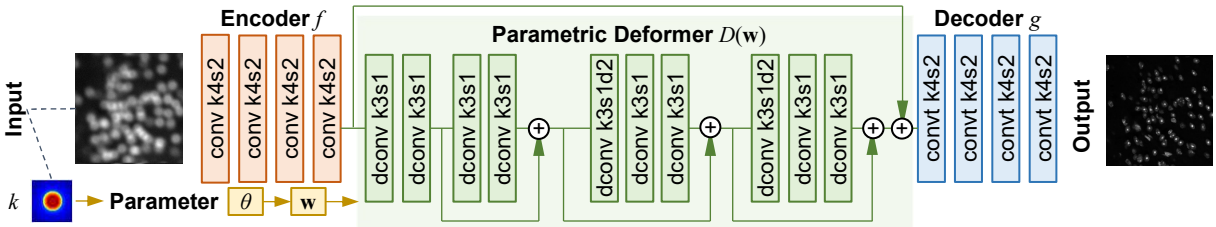

Figure 1. Architecture of the latent space deformation network. For brevity, nonlinear activations, batch normalizations, and hidden layers to encode the parameter w are not shown. `conv`, `convt` and `dconv` stand for convolution layer, convolution transpose layer, and dynamic convolution, respectively. `k3s1d2` means kernel size 3, stride 1, dilation 2. All convolutions are 1-dilated unless otherwise noted.

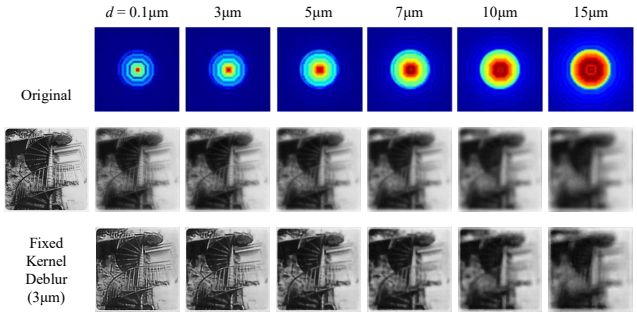

Figure 2. An example set of NBDs with varying kernels. Top: six kernels with different defocal distances. All kernels are scalar real-valued intensity maps, colorized with false color for visualization purposes. Center: the original clean image and the blurred images from the kernels above. Bottom: an encoder-decoder deblurring network trained with the 3μm kernel fails to deblur images from other kernels.

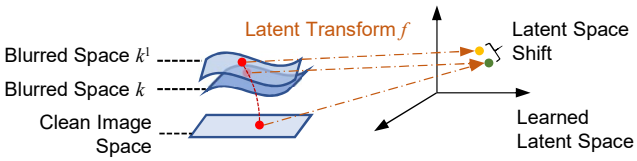

Figure 3. Shift in latent space when encoding a blurred image from another kernel.

words, we establish a shared latent space, where both noisy and clean images reside, and the denoising transform.

While many deblurring and super-resolution methods verified that this shared latent space can be found with partial (corrupted) observations, many of them assume that the underlying corruption is fairly similar, e.g. simple downsizing. When blurred images are generated with different blurring kernels, as in Fig. 2, a simple encoder-decoder trained on one kernel cannot adapt to all different cases.

The reason for this degradation is that blurred images under different kernels do not lie in the same manifold, as Fig. 3 illustrates. From the training, the shared latent space of clean images and blurred images from the kernel k is established, but it is not shared with a new kernel k^1; using the learned transform on a blurred image from k^1 would result in a shifted point in a deformed latent space. Now consider the latent space with k as a standard, or *reference latent space*. The deformation from the reference space can be modeled as a parametric vector field, parameterized by k^1. Thus, our objective is to learn the deformation with a range of $\{k^j\}$ so that the deblurring model works with more different kernels.

3.3. Network Design

We propose to train this deformable latent space in two stages. At the first stage, we establish the reference latent space. With a "standard" blurring kernel $k = k^0$, we draw clean-blurred image pairs $\{\mathbf{I}_i, \mathbf{B}_i\}$ to train a deblurring autoencoder

$$\mathbf{h}_i = f(\mathbf{B}_i), \quad \mathbf{O}_i = g(\mathbf{h}_i), \tag{2}$$

where f is the encoder, g is the decoder, $\mathbf{B}_i$ is the blurred input, $\mathbf{O}_i$ is the output, and $\mathbf{h}_i$ is the latent encoded vector. The training objective is to fit the decoder output $\mathbf{O}_i$ to the clean image $\mathbf{I}_i$

$$\arg\min_{f,g} \sum_{i=1}^{N} \mathcal{L}((g \circ f)(\mathbf{B}_i), \mathbf{I}_i), \tag{3}$$

where N is the number of training images, $(g \circ f)(\cdot)$ means $g(f(\cdot))$, and $\mathcal{L}$ is a loss function, *e.g.* mean squared error (MSE).

Once $\{f, g\}$ are trained, we move onto the second stage. We may encode blurred images generated from new kernels using Eqn. 2, but the encoded vectors will shift in the reference space. With a new blurred image $\mathbf{B}_i^j$ from kernel k^j, we freeze $\{f, g\}$ and compute a deformer D in the middle of the encoder-decoder:

$$\mathbf{h}_i^j = f(\mathbf{B}_i^j), \quad \mathbf{h}_i = D(\mathbf{h}_i^j, \theta^j), \quad \mathbf{O}_i = g(\mathbf{h}_i). \tag{4}$$

Kernel parameter θ^j directly governs D to rectify the shift in latent space. We will explain the parameterization in Section 3.4. For training, we draw both the "standard" blurred image $\mathbf{B}_i$ and new blurred images $\mathbf{B}_i^j$ based on the same clean image $\mathbf{I}_i$ to compute latent space deformation. The

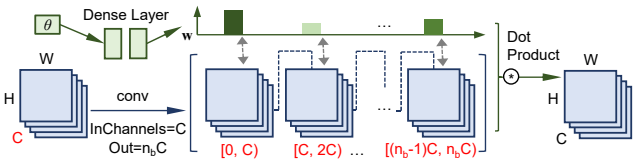

Figure 4. Structure of a dynamic convolution. The middle tensor in brackets $[\cdots]$ has $n_b C$ channels as a result of the convolution, which are spread out as n_b groups to take the dot product as described in Eqn. 8.

training objective is to deform the non-standard latent feature vectors to fit in the reference space

$$\arg\min_{D} \sum_{i=1}^{N} \sum_{j=0}^{M} \mathcal{L}(f(\mathbf{B}_i), (D \circ f)(\mathbf{B}_i^j, \theta_i^j)), \qquad (5)$$

where N is the number of training images, M is the number of new kernels, excluding k^0. The complete network consists of the encoder, decoder and deformer $\{f, g, D\}$.

3.4. Parameterization with Dynamic Convolution

The latent space deformer D is parameterized by θ^j. As we intend to learn an additive "shift" to rectify the latent space vector, we model D as a residual subnet as follows

$$D(\mathbf{h}_i^j, \theta^j) = \mathbf{h}_i^j + R(\mathbf{h}_i^j, \theta^j). \qquad (6)$$

As illustrated in Fig. 1, the residual subnet consists of four residual blocks of two convolutional layers, but the latter two blocks each have a preceding 2-dilated convolution to expand the receptive field.

Dynamic Convolution. To parameterize the network, we compute dynamic convolutions [40] in place of all the convolutions in the deformer D. Simply put, a conventional 2D convolution

$$\mathrm{Conv2d}(\mathbf{h}) = \mathbf{K} * \mathbf{h} \qquad (7)$$

has a filter $\mathbf{K}$ of shape $d_{\text{in}} \times d_{\text{out}} \times d_h \times d_w$, where d_{in} is the number of input channels, d_{out} is the number of output channels and d_h, d_w are the filter height and width. Dynamic convolution expands it to a group of n_b such filters $\{\mathbf{K}_b\}$, like an n_b-dimensional basis. Taking an input feature $\mathbf{h}$ and a n_b-dimensional basis coefficient $\mathbf{w} = (w_1, w_2, \ldots, w_{n_b})$, dynamic convolution computes the dot product of the n_b convolutions (Fig. 4)

$$\mathrm{DConv2d}(\mathbf{h}, \mathbf{w}) = \sum_{b=1}^{n_b} w_b \cdot (\mathbf{K}_b * \mathbf{h}). \qquad (8)$$

Basis Coefficient Estimator. The basis coefficient $\mathbf{w}^j$ is an embedding of kernel parameter θ^j. We use multi-layer perceptrons (MLPs) to encode θ^j and normalize it to unit L_2 norm

$$\mathbf{w}_{\text{raw}}^j = \mathrm{MLP}(\theta^j), \quad \mathbf{w}^j = \mathbf{w}_{\text{raw}}^j / \|\mathbf{w}_{\text{raw}}^j\|_2. \qquad (9)$$

Alternatively softmax can be used too:

$$\mathbf{w}^j = \mathrm{softmax}(\mathbf{w}_{\text{raw}}^j). \qquad (10)$$

Note that each dynamic convolution is associated with a basis coefficient MLP. These MLPs can be separate or share weights. We share weights among each convolutional block, *i.e.* layer groups $\{1, 2\}, \{3, 4\}, \{5, 6, 7\}, \{8, 9, 10\}$ in D each use one shared MLP.

Alternate Training. The learnable parameters of the dynamic convolutions can be divided into two components: the filter group $\{\mathbf{K}_b\}$, which is a feature "basis", and the "basis coefficient" estimator $\mathrm{MLP}(\cdot)$. To facilitate the training, we alternate the parameter updates. For each epoch, we go through half of the training batches only to optimize the basis $\{\mathbf{K}_b\}$, and keep the coefficients $\mathrm{MLP}(\cdot)$ unchanged; and use the other half to only update $\mathrm{MLP}(\cdot)$, while freezing $\{\mathbf{K}_b\}$.

4. Experiments and Evaluations

In this section, we report our main evaluation results. See supplementary material for the fully executable kernel formulations and additional output samples.

4.1. Dataset Preparation

First we describe the kernel selections for different blurred images in our learning dataset. Based on simulated probes of zone plates used in ptychography, we first generate six 64×64 probes with defocal distances $d = \{0.1, 3, 5, 7, 10, 15\}\mu\mathrm{m}$ respectively, then multiply them with a band-limited filter to restrict the range of the probes to 0.35 of the radius. Since the defocal distance d is the sole factor that controls the variations of probes in our experiment, we set it as the kernel parameter $\theta^j = d$ that we are going to feed in the deformation network.

We use both natural image data and real fluorescence data in our experiments, because natural images offer more variety and larger amounts of images for deep CNN training. For natural images, We use the validation images from ILSVRC 2012 [22] to generate the NBD learning dataset. We randomly take 10,000 images to make the training set, and 100 images to make the testing set. All images are resized to 256×256 and normalized to $[0, 1]$. The images are convolved with each one of these different kernels, and added with Gaussian noise with a variance of 0.02, to generate all different blurred versions. The kernels and blurred images are shown in Fig. 2. We can see the kernels have different ranges as well as unique patterns within range.

For fluorescence images, we take the clean images from the FMD dataset [39] to generate our NBD data. We randomly take 2,000 images to make the training set, and 40 images to make the testing set. We use the same blurring kernels and processing as in natural image data to generate the full dataset.

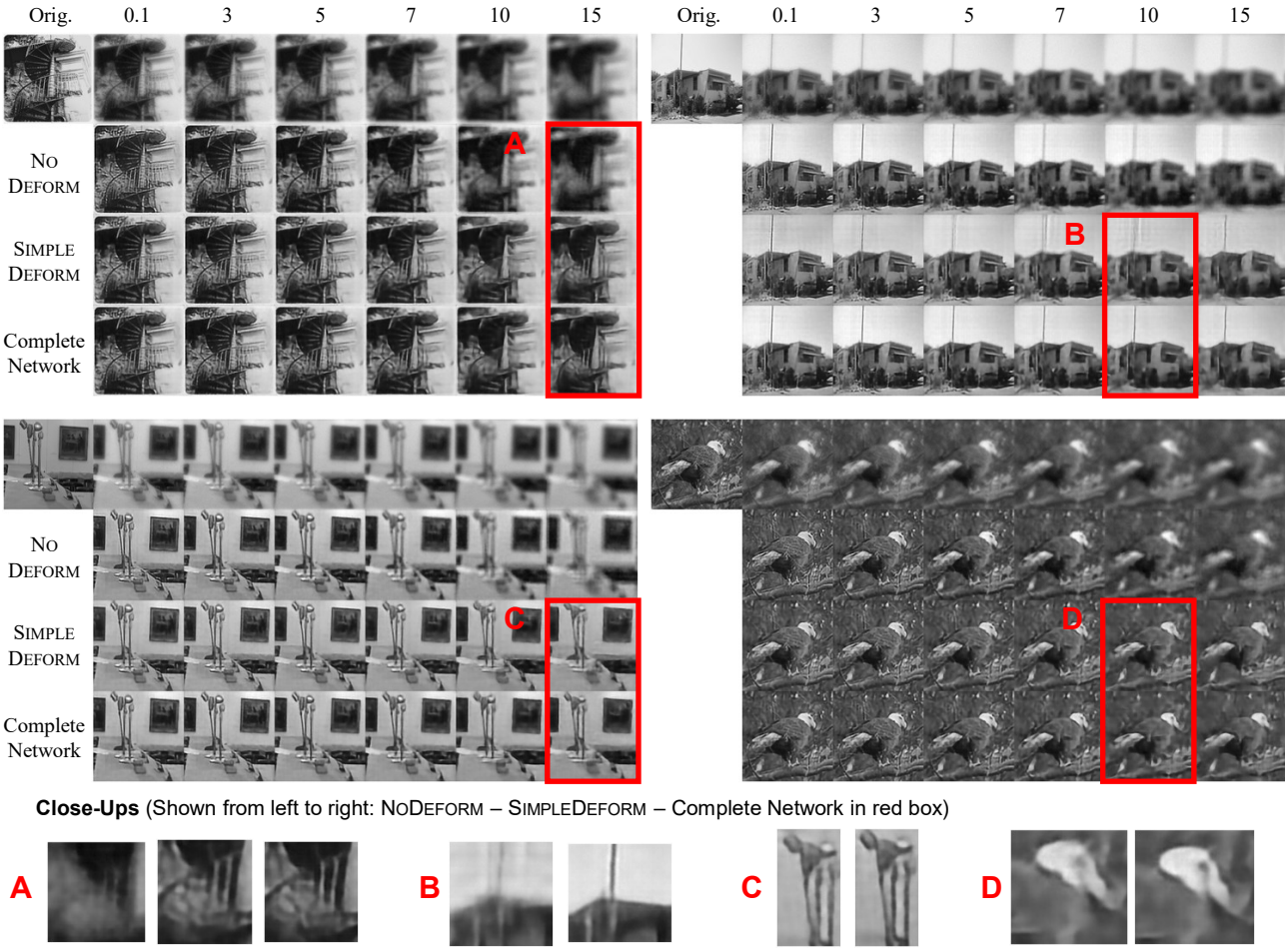

Figure 5. Comparison of NODEFORM, SIMPLEDEFORM and the complete deformation network.

4.2. Comparisons of Deformers

We verify the effect of the parametric deformer. First, we test the reconstruction without latent space deformation. We only use the encoder-decoder $\{f, g\}$ to deblur images from all kernels – we call this NODEFORM.

To further demonstrate the benefit of fully parameterizing every deformer layer, we compare it with a contrived setup with less interaction with the parameters – we call this SIMPLEDEFORM. Based on our deformer in Section 3.4, we replace all the dynamic convolutions with regular convolutions. After the final 256-channel feature is acquired, we reweight it by multiplying a 256-dimensional vector $\mathbf{w}$, which is computed using a 1-16-256 MLP with defocal distance d

$$\mathbf{w} = \mathrm{MLP}(d), \tag{11}$$

and finally an additional BatchNorm-ReLU-Conv is performed. This gives a one-time engagement with the parameters. In stark contrast to this artificial example, the parameter θ^j is deeply involved in every layer in our model. Also, for the parameter embedding $\mathbf{w}^j$, we make it different at each convolutional block to fully tailor the deformer.

To train the deformer networks, we use the 3μm kernel to train the reference latent space. We use the ADAM optimizer [10] with learning rate 0.0002, and train them on natural image dataset. We train the first stage ($\{f, g\}$) for 20 epochs, and the second stage (D) for 30 epochs, 50 epochs in total.

The results of all these methods are shown in Fig. 5 and Table 1. We can see that SIMPLEDEFORM and the complete deformer method both show visible improvements over NODEFORM. From the close-up details, the complete deformer is more expressive than SIMPLEDEFORM.

	ND	SD	Comp.
Mean PSNR (dB)	20.1426	21.0724	21.2795

Table 1. Mean PSNR of NODEFORM (ND), SIMPLEDEFORM (SD) and the complete deformation network (Comp.).

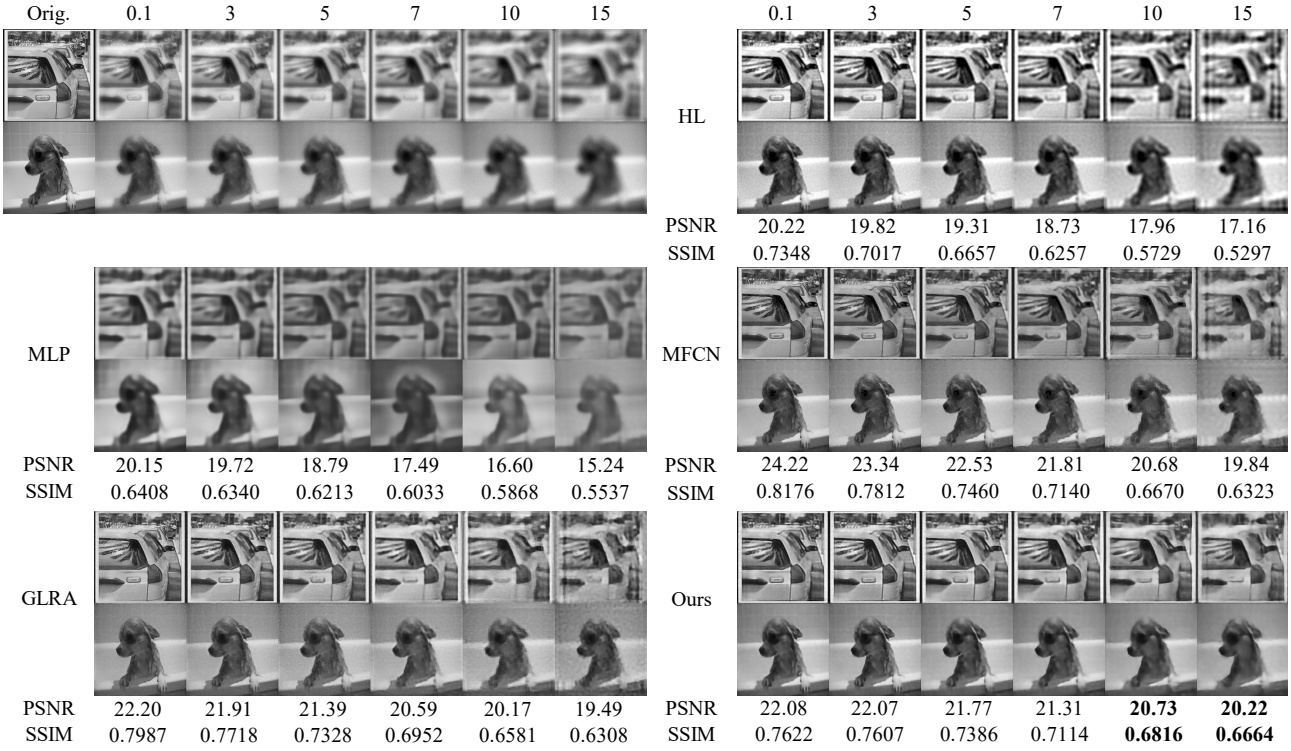

Figure 6. Comparisons of different NBD methods. Here we report the mean PSNR and mean SSIM per kernel on the testing set.

4.3. Comparisons with Existing NBD Methods

We compare our method with existing NBD methods: Hyper-Laplacian (HL) [11], MLP [26], multi-input fully convolutional network (MFCN) [30], and generalized low-rank approximation (GLRA) [18]. We use the natural image dataset to train all these networks. The corresponding kernels are given to all the algorithms. HL is evaluated with the parameters $\alpha = 2/3, \lambda = 2 \times 10^3$, and MFCN takes HL results with $\alpha = 2/3, \lambda = \{2 \times 10^2, 2 \times 10^3, 2 \times 10^4\}$ as input. We train MLP, MFCN and GLRA all for 50 epochs. The results are shown in Fig. 6.

We can see from the comparisons that our method produces less artifacts than all these methods visually, and outperforms HL, MLP and GLRA in terms of mean PSNR and SSIM in most of the kernel settings. While MFCN has higher mean PSNR and SSIM for some kernels, it has visibly more rippling artifacts.

From these results, we argue that our deformer in the low-dimensional latent space has two advantages. First, there is less artifact from approximation. In contrast, *e.g.* MFCN is fully convolutional. It deblurs the images with a few different optimization parameter settings – all of which are suboptimal and contain artifact – and then fuses them all with a deep network. Despite the intensive training, the remnants of artifact from the base deblurred images are not very well eliminated. Second, our method is lighter and faster to compute. We will show this later in Section 4.5.

4.4. Comparisons with Deep BD Methods

We compare our method with a few deep BD methods: Nah *et al.*'s Deep Multi-scale CNN (DeepDeblur) [14], Tao *et al.*'s Scale-recurrent Network (SRN) [29], and Ye *et al.*'s Scale-Iterative Upscaling Network (SIUN) [36]. We use the natural image dataset to train these networks. The results are shown in Figure 7. We can see that SRN and SIUN do not learn effectively from this diversely blurred dataset, which demonstrates that the NBD problem needs to be addressed specifically, and general deep deblurring methods are likely not suitable. DeepDeblur has better mean PSNR and SSIM than our method, but note that all these deep BD methods are multi-scale networks that are much deeper and take much longer to train one epoch.

4.5. Comparison of Running Times

Our deformation network computes feature transforms in the low dimensional feature space, which is much faster to run compared to fully convolutional networks that compute full size tensors. Below are the running times of training the following network on the natural image dataset, for one epoch, on an NVIDIA GeForce RTX 2080 Ti (See Table 2).

4.6. Results on Fluorescence Data

We also use real fluorescence data to compare the NBD methods reported in Section 4.3: HL, MLP, MFCN and

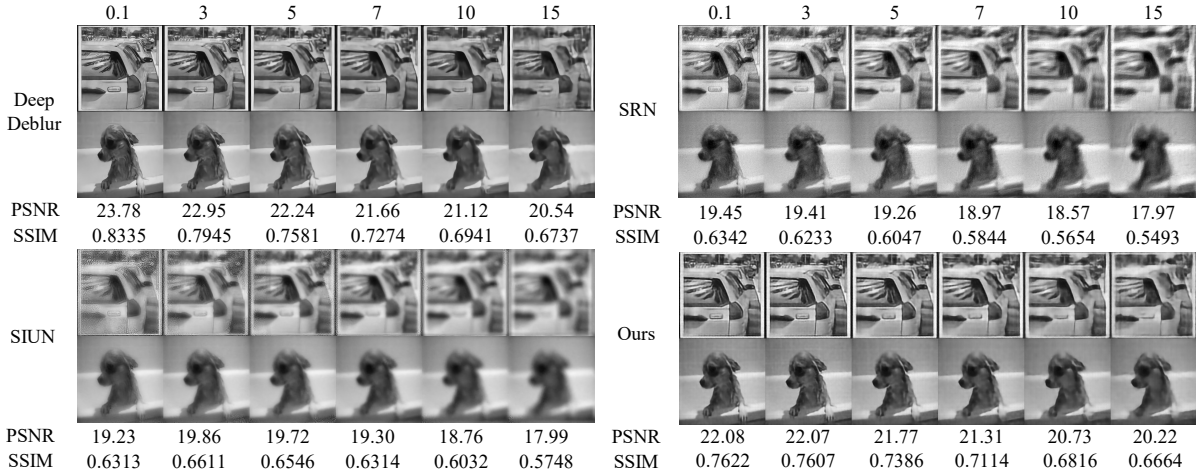

Figure 7. Comparisons with deep BD methods. Here we report the mean PSNR and mean SSIM per kernel on the testing set. Ground truth and blurred input are the same as shown in Figure 6.

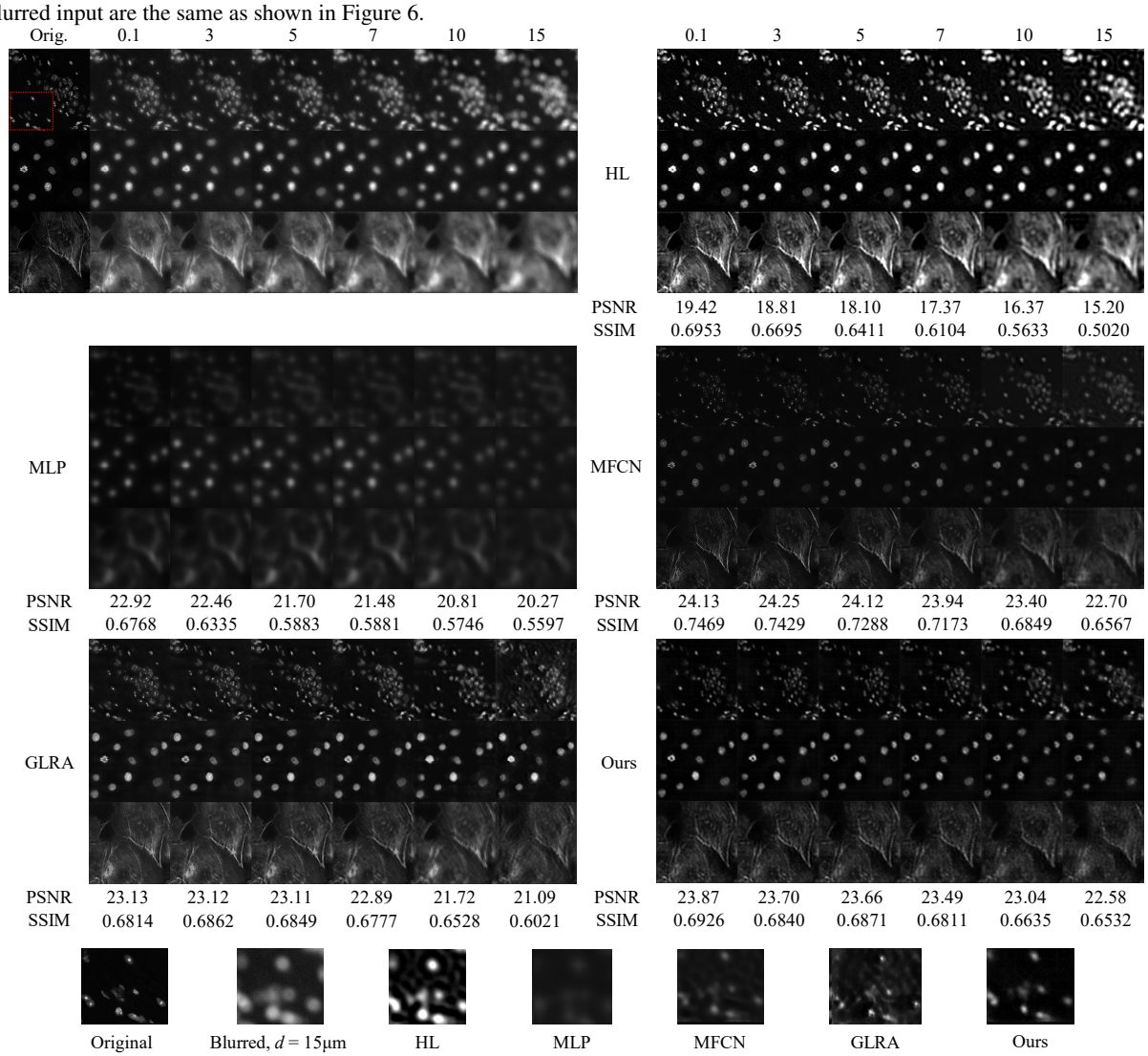

Figure 8. Comparisons of different NBD methods on real fluorescence data. Close-ups of deblurring results of the first image, with kernel $d = 15\mu$m, shown on the bottom. Here we report the mean PSNR and mean SSIM per kernel on the testing set.

Architecture	Time
Ours, Stage 1, Encoder-Decoder	0:00:27
Ours, Stage 2, Deformer	0:07:51
MFCN	3:18:33
GLRA	0:42:18
DeepDeblur	3:04:18
SRN	0:44:19
SIUN	1:04:28

Table 2. Time to train for one epoch for different neural networks.

GLRA. For this dataset, our latent space deformation is trained 80 epochs for the first stage ($\{f, g\}$), and 120 epochs for the second stage (D). The other methods are all trained for 200 epochs. The results are shown in Fig. 8. Our deformation method produces fewer artifacts in more severely blurred cases.

5. Conclusions and Future Work

In this paper, we proposed a latent space deformation approach to solve parametric image non-blind deblurring. The deformer network we developed is empirically effective in manipulating the compact latent feature representation, and operating in a relatively low-dimensional latent space is fast and light-weight compared to full-size convolutional networks.

We designed our method with emphasis specifically on modeling the varying blurring kernel K, because it is crucial for the adaptability of the method, which is illustrated by experiment performance with a drastically different set of kernels. With reliable NBD parameterized by K, we will be able to perform simultaneous ptychography and fluorescent imaging acquisition with high-resolution and high-throughput to advance scientific discoveries in energy materials, bio-materials, and beyond.

For future work, we are going to build a stronger deformer with techniques such as attention mechanism and adversarial learning to warp latent features closer to the reference space. On the other hand, we plan to extend the idea of parametric latent space to bivariate and multivariable parametric spaces to model more variety of the blurring kernels. We also plan to use deep neural networks to extract semantic parameters from the kernel itself, and generalize this parameterized generative approach to other imaging problems.

Acknowledgements. This work was supported by NSF IIS-1715985 and NSF IIS-1812606. This research used resources of the Center for Functional Nanomaterials (CFN), and the National Synchrotron Light Source (NSLS) II. CFN and NSLS-II are U.S. Department of Energy (DOE) Office of Science Facilities operated for the DOE Office of Science by Brookhaven National Laboratory under Contract No. DE-SC0012704.

References

[1] S. Cho, J. Wang, and S. Lee. Handling outliers in non-blind image deconvolution. In *ICCV*, pages 495–502. IEEE, 2011.

[2] A. Danielyan, V. Katkovnik, and K. Egiazarian. BM3D frames and variational image deblurring. *IEEE TIP*, 21(4):1715–1728, 2011.

[3] C. Dong, C. C. Loy, K. He, and X. Tang. Image super-resolution using deep convolutional networks. *IEEE TPAMI*, 38(2):295–307, 2015.

[4] J. Dong, J. Pan, D. Sun, Z. Su, and M.-H. Yang. Learning data terms for non-blind deblurring. In *ECCV*, pages 748–763, 2018.

[5] S. Ehn, F. M. Epple, A. Fehringer, D. Pennicard, H. Graafsma, P. Noël, and F. Pfeiffer. X-ray deconvolution microscopy. *Biomed. Opt. Express*, 7(4):1227–1239, 2016.

[6] H. Faulkner and J. Rodenburg. Movable aperture lensless transmission microscopy: a novel phase retrieval algorithm. *Phys. Rev. Lett.*, 93(2):023903, 2004.

[7] H. Hotelling. Relations Between Two Sets of Variates. *Biometrika*, 28(3/4):321–377, 1936.

[8] J. Ihli, R. R. Jacob, M. Holler, M. Guizar-Sicairos, A. Diaz, J. C. Da Silva, D. F. Sanchez, F. Krumeich, D. Grolimund, M. Taddei, et al. A three-dimensional view of structural changes caused by deactivation of fluid catalytic cracking catalysts. *Nature Communications*, 8(1):1–10, 2017.

[9] H. Ji and K. Wang. Robust image deblurring with an inaccurate blur kernel. *IEEE TIP*, 21(4):1624–1634, 2011.

[10] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[11] D. Krishnan and R. Fergus. Fast image deconvolution using hyper-Laplacian priors. In *NeurIPS*, pages 1033–1041, 2009.

[12] J. Kruse, C. Rother, and U. Schmidt. Learning to push the limits of efficient fft-based image deconvolution. In *ICCV*, pages 4586–4594, 2017.

[13] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean. Distributed representations of words and phrases and their compositionality. In *NeurIPS*, pages 3111–3119, 2013.

[14] S. Nah, T. H. Kim, and K. M. Lee. Deep multi-scale convolutional neural network for dynamic scene deblurring. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3883–3891, 2017.

[15] J. Ngiam, A. Khosla, M. Kim, J. Nam, H. Lee, and A. Y. Ng. Multimodal Deep Learning. In *ICML*, pages 689–696, 2011.

[16] D. Pathak, P. Krahenbuhl, J. Donahue, T. Darrell, and A. A. Efros. Context encoders: Feature learning by inpainting. In *CVPR*, pages 2536–2544, 2016.

[17] D. Ren, W. Zuo, D. Zhang, J. Xu, and L. Zhang. Partial deconvolution with inaccurate blur kernel. *IEEE TIP*, 27(1):511–524, 2017.

[18] W. Ren, J. Zhang, L. Ma, J. Pan, X. Cao, W. Zuo, W. Liu, and M.-H. Yang. Deep non-blind deconvolution via generalized low-rank approximation. In *NeurIPS*, pages 297–307, 2018.

[19] W. H. Richardson. Bayesian-based iterative method of image restoration. *J. Opt. Soc. Am.*, 62(1):55–59, 1972.

[20] J. Rodenburg, A. Hurst, A. Cullis, B. Dobson, F. Pfeiffer, O. Bunk, C. David, K. Jefimovs, and I. Johnson. Hard-x-ray lensless imaging of extended objects. *Phys. Rev. Lett.*, 98(3):034801, 2007.

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

[22] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *IJCV*, 115(3):211–252, 2015.

[23] P. Sarder and A. Nehorai. Deconvolution methods for 3-d fluorescence microscopy images. *IEEE Signal Process. Mag.*, 23(3):32–45, 2006.

[24] U. Schmidt and S. Roth. Shrinkage fields for effective image restoration. In *CVPR*, pages 2774–2781, 2014.

[25] U. Schmidt, C. Rother, S. Nowozin, J. Jancsary, and S. Roth. Discriminative non-blind deblurring. In *CVPR*, pages 604–611, 2013.

[26] C. J. Schuler, H. Christopher Burger, S. Harmeling, and B. Scholkopf. A machine learning approach for non-blind image deconvolution. In *CVPR*, pages 1067–1074, 2013.

[27] J.-B. Sibarita. Deconvolution microscopy. In *Microscopy Techniques*, pages 201–243. Springer, 2005.

[28] L. Sun, S. Cho, J. Wang, and J. Hays. Good image priors for non-blind deconvolution. In *ECCV*, pages 231–246. Springer, 2014.

[29] X. Tao, H. Gao, X. Shen, J. Wang, and J. Jia. Scale-recurrent network for deep image deblurring. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8174–8182, 2018.

[30] S. Vasu, V. Reddy Maligireddy, and A. Rajagopalan. Non-blind deblurring: Handling kernel uncertainty with CNNs. In *CVPR*, pages 3272–3281, 2018.

[31] P. Vincent, H. Larochelle, Y. Bengio, and P.-A. Manzagol. Extracting and composing robust features with denoising autoencoders. In *ICML*, pages 1096–1103, 2008.

[32] K. Watanabe, Y. Kotaka, N. Nakanishi, T. Yamazaki, I. Hashimoto, and M. Shiojiri. Deconvolution processing of HAADF STEM images. *Ultramicroscopy*, 92(3-4):191–199, 2002.

[33] N. Wiener. *Extrapolation, interpolation, and smoothing of stationary time series*. The MIT press, 1964.

[34] L. Xu, J. S. Ren, C. Liu, and J. Jia. Deep convolutional neural network for image deconvolution. In *NeurIPS*, pages 1790–1798, 2014.

[35] Y. Yan, W. Ren, Y. Guo, R. Wang, and X. Cao. Image deblurring via extreme channels prior. In *CVPR*, pages 4003–4011, 2017.

[36] M. Ye, D. Lyu, and G. Chen. Scale-iterative upscaling network for image deblurring. *IEEE Access*, 8:18316–18325, 2020.

[37] J. Zhang, J. Pan, W.-S. Lai, R. W. Lau, and M.-H. Yang. Learning fully convolutional networks for iterative non-blind deconvolution. In *CVPR*, pages 3817–3825, 2017.

[38] K. Zhang, W. Zuo, S. Gu, and L. Zhang. Learning deep CNN denoiser prior for image restoration. In *CVPR*, pages 3929–3938, 2017.

[39] Y. Zhang, Y. Zhu, E. Nichols, Q. Wang, S. Zhang, C. Smith, and S. Howard. A Poisson-Gaussian denoising dataset with real fluorescence microscopy images. In *CVPR*, pages 11710–11718, 2019.

[40] F. Zhao, J. Zhao, S. Yan, and J. Feng. Dynamic conditional networks for few-shot learning. In *ECCV*, pages 19–35, 2018.

[41] D. Zoran and Y. Weiss. From learning models of natural image patches to whole image restoration. In *ICCV*, pages 479–486. IEEE, 2011.