Self-supervised Blind Motion Deblurring with Deep Expectation Maximization

Ji Li, Weixi Wang, Yuesong Nan, and Hui Ji
Department of Mathematics, National University of Singapore, 119076, Singapore
matliji@nus.edu.sg, wangweixi@u.nus.edu, nanyuesong@u.nus.edu, matjh@nus.edu.sg

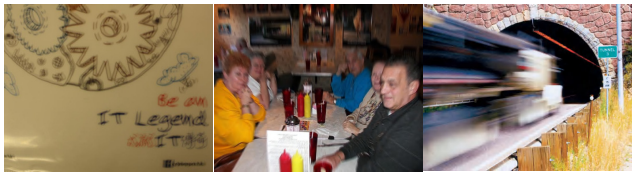
Abstract

When taking a picture, any camera shake during the shutter time can result in a blurred image. Recovering a sharp image from the one blurred by camera shake is a challenging yet important problem. Most existing deep learning methods use supervised learning to train a deep neural network (DNN) on a dataset of many pairs of blurred/latent images. In contrast, this paper presents a dataset-free deep learning method for removing uniform and non-uniform blur effects from images of static scenes. Our method involves a DNN-based re-parametrization of the latent image, and we propose a Monte Carlo Expectation Maximization (MCEM) approach to train the DNN without requiring any latent images. The Monte Carlo simulation is implemented via Langevin dynamics. Experiments showed that the proposed method outperforms existing methods significantly in removing motion blur from images of static scenes.

1. Introduction

Motion blur occurs when the camera shakes during the shutter time, resulting in a blurring effect. Blur is uniform when the scene depth is constant and moves along the image plane. For other camera movements, the blur is non-uniform. In dynamic scenes with moving objects, the blur is also non-uniform. Different types of motion blur are illustrated in Figure 1. This paper aims to address the problem of removing uniform and non-uniform motion blur caused by camera shake from an image. Removing motion blur from an image is a blind deblurring problem. It is a challenging task as it requires estimating two unknowns the latent image and blurring operator from a single input.

Deep learning, particularly supervised learning, has recently emerged as a powerful tool for solving various image restoration problems, including blind deblurring. Many of these works rely on supervised learning, as seen in *e.g.* [1–15]. Typically, these supervised deep learning methods train a deep neural network (DNN) on a large number of training samples, which consist of pairs of latent/blur images. Furthermore, to address general blurring effects, most methods take a physics-free approach. In other words, these methods directly learn a model that maps a blurred image to a latent image without using any prior information about the blurring process.



(a) (b) (c)

Figure 1. Different motion-blurring effects. (a)–(b) Uniform and non-uniform blurring caused by camera shake; (c) Non-uniform blurring of the dynamic scene (not addressed in this paper).

The advantage of a physics-free supervised learning method is its ability to handle many types of motion blur effects. However, it has a significant disadvantage: to achieve good generalization, the training dataset must cover all motion-blurring effects. Because motion blur is determined by both 3D camera motion (six parameters) and scene depth, which can vary significantly among images, an enormous number of training samples are required to adequately cover the motion blur effects. This task can be very costly and challenging. One possible solution is to synthesize blurred images. However, as shown in [16], a model trained on samples synthesized using existing techniques (*e.g.* [17]) does not generalize well to real-world images.

Some approaches consider the physics of motion blur. Phong *et al.* [18] proposed learning a family of blurring operators in an encoded blur kernel space, and Li *et al.* [19] proposed learning a more general class of degradation operators from input images. However, these physics-aware methods also rely on supervised learning and thus face the same dataset limitations as the physics-free methods.

1.1. Discussion on existing dataset-free methods

Motivated by the challenge of practical data collection, there is a growing interest in relaxing the requirement for training data when developing deep learning solutions for motion deblurring. Some approaches require spe-

cific data acquisition, such as multiple frames of the same scene [20], while others are semi-supervised, relying on unpaired training samples with ground truth images for training a GAN [21]. There are also a few works on dataset-free deep learning methods for uniform blind image deblurring; see *e.g.* [22–24].

When training a DNN to deblur an image without seeing ground truth images, it is necessary to incorporate prior knowledge of the physics of motion blur. However, existing dataset-free methods [22] for blind deblurring are limited to uniform blur, where the blur process is modeled by a convolution: $\boldsymbol{g} = \boldsymbol{k} \otimes \boldsymbol{f}$, where $\boldsymbol{f}$ denotes the latent image, $\boldsymbol{g}$ denotes the input, and $\boldsymbol{k}$ denotes the blur kernel. Uniform motion blur only occurs when the scene depth is constant and camera motion is limited to in-image translation, making it not applicable to more complex camera motion. Moreover, these methods have a lot of room for improvement, as they do not achieve competitive performance compared to state-of-the-art non-learning methods.

1.2. Main idea

In this paper, our goal is to develop a dataset-free deep learning method for removing motion blur from a single image, caused by general camera shake. Similar to existing dataset-free methods, when training a DNN to deblur an image without seeing any truth image, some prior knowledge about the physics of motion blur needs to be utilized. In this paper, we limit our study to recovering images of static scenes without any moving objects. In our proposed approach, we utilize the so-called *space-variant overlap-add* (SVOLA) formulation [25, 26] to model motion blur of static scenes. This formulation describes the relationship between a blurred image $\boldsymbol{g}$ and its corresponding latent image $\boldsymbol{f}$ as follows:

$$\boldsymbol{g} = F(\boldsymbol{f}, \boldsymbol{K}) + \boldsymbol{n} = \sum_{i=1}^{P} \boldsymbol{k}_i \otimes (\boldsymbol{w}(\cdot - c_i) \odot \mathcal{P}_i \boldsymbol{f}) + \boldsymbol{n}, \quad (1)$$

Here, $\odot$ denotes entry-wise multiplication, $\otimes$ denotes convolution, and $\mathcal{P}_i$ is a mask operator that extracts the i-th patch from the image. $\boldsymbol{k}_i$ is the i-th kernel and $\boldsymbol{w}(\cdot - c_i)$ is a window function that is translated to align with the center c_i of the i-th image patch. The window function $\boldsymbol{w}$ is normalized such that $\sum_{i=1}^{P} \boldsymbol{w}(\cdot - c_i) = \mathbf{1}$, for example, using the 2D Modified Bartlett-Hanning window [27]. When all P kernels $\{\boldsymbol{k}_i\}_{i=1}^{P}$ are the same, the SVOLA model degenerates to the case of uniform blurring:

$$\boldsymbol{g} = \boldsymbol{k} \otimes \boldsymbol{f}. \quad (2)$$

For an SVOLA-based model, there are two unknowns: the latent image $\boldsymbol{f}$ and the kernel set $\boldsymbol{K} = \{\boldsymbol{k}_j\}_{j=1}^{P}$.

Similar to existing works, such as Double-DIP for image decomposition and Ren *et al*. for uniform deblurring, we re-parameterize the latent image and kernel set using two DNNs. This DNN-based re-parametrization is motivated by the implicit prior induced by convolutional neural networks (CNNs), known as the deep image prior (DIP). However, the regularization effect induced by DIP alone is not sufficient to avoid likely overfitting. One approach is to introduce additional regularization drawn inspiration from traditional non-learning methods.

Discussion on MAP-relating methods. For simplicity, consider the case of uniform blur where $\boldsymbol{g} = \boldsymbol{k} \otimes \boldsymbol{f}$. As the ML (maximum likelihood) estimator of the pair $(\boldsymbol{k}, \boldsymbol{f})$ by

$$\max_{\boldsymbol{k},\boldsymbol{f}} \log p(\boldsymbol{g}|\boldsymbol{k}, \boldsymbol{f}) \quad (3)$$

does not resolve solution ambiguity of blind deblurring, most non-learning methods are based on the maximum a posteriori (MAP) estimator, which estimates $(\boldsymbol{k}, \boldsymbol{f})$ by

$$\max_{\boldsymbol{k},\boldsymbol{f}} \log p(\boldsymbol{k}, \boldsymbol{f}|\boldsymbol{g}) = \min_{\boldsymbol{k},\boldsymbol{f}} -\log p(\boldsymbol{g}|\boldsymbol{k}, \boldsymbol{f}) - \log p(\boldsymbol{f}) - \log p(\boldsymbol{k}).$$

An MAP estimator requires the definition of two prior distributions: $p(\boldsymbol{k})$ and $p(\boldsymbol{f})$. A commonly used prior distribution for motion deblurring assumes that $\boldsymbol{f}$ follows a Laplacian distribution: $\log p(\boldsymbol{f}) \propto -\|\nabla \boldsymbol{f}\|_1$, also known as total variation (TV) regularization. Such a TV-based MAP estimator is proposed in [22] for blind uniform deblurring.

There are two concerns about a TV-relating MAP estimator. One is the pre-defined TV regularization for latent images limits the benefit of data adaptivity brought by a DNN. The other is the possible convergence to an incorrect local minimum far away from the truth $(\boldsymbol{f}, \boldsymbol{k})$ or even degenerated trivial solution $(\boldsymbol{g}, \boldsymbol{\delta})$. Indeed, the second issue has been extensively discussed in existing works; see *e.g.* [28–30].

From MAP estimator of $(\boldsymbol{f}, \boldsymbol{K})$ to EM algorithm for ML estimator of $\boldsymbol{K}$. Besides MAP, many other statistical inference schemes have also been successfully used for blind uniform deblurring, *e.g.* variational Bayesian inference [30, 31]; and EM algorithm [32]. EM is an iterative scheme to find maximum likelihood (ML) estimate with the introduction of latent variables. For blind deblurring, EM aims at finding ML estimate of the marginal likelihood of the unknown parameter $\boldsymbol{K}$ only by marginalizing over the image $\boldsymbol{f}$ (latent variable).

Our approach. Inspired by the effectiveness of the EM algorithm and marginal likelihood optimization for uniform deblurring in terms of performance and stability, we propose to use the EM algorithm as a guide to develop a self-supervised learning approach. Specifically, we introduce a dataset-free deep learning method for both uniform and non-uniform blind motion deblurring, which is based on the Monte Carlo expectation maximization (MCEM) algorithm. In summary, our method is built upon the efficient EM algorithm in DNN-based representation of latent image and blurring operator.

1.3. Main contribution

In this paper, we present a self-supervised deep learning approach for restoring motion-blurred images. Our main contributions can be summarized as follows:

1. The first dataset-free deep learning method for removing general motion blur (uniform and non-uniform) from images due to camera shake. To our knowledge, all existing dataset-free methods are limited to uniform motion blur.
2. The first approach that combines DNN-based reparametrization and EM algorithm, bridging the gap between classical non-learning algorithms and deep learning. The proposed MCEM-based deep learning method can see its applications in other image recovery tasks.
3. A powerful method that significantly outperforms existing solutions for blind motion deblurring. Our method demonstrates superior performance in recovering images affected by both uniform and non-uniform motion blur.

2. Related works

Non-learning methods for motion deblurring. Most existing non-learning methods focused on uniform motion and rely on the concept of the MAP estimator, which involves defining priors on the image and kernel. For instance, sparsity priors based on image gradients [33–36] or wavelet transforms [37, 38], image-patch recurrence priors [39, 40], and dark/extreme channel priors [41, 42]. Another approach relies on the *variational Bayesian inference* of the blur kernel, where the latent image is considered as a hidden variable [28, 30, 31, 43]. To address the computational issue of the posterior distribution of the latent image, some methods rely on certain conjugate probability models of the distribution [30, 31, 43]. Other methods, such as the *VEM* algorithms *e.g*. [29, 32], approximate the posterior distribution of image gradients using a normal distribution.

The literature on non-uniform motion deblurring mainly focuses on modeling the non-uniform blurring effect. Ji and Wang [44] proposed a piece-wise convolution model, while Tai *et al*. [45] proposed a homography-based model, which was later simplified in [46, 47]. Harmeling *et al*. [25] presented an SVOLA model in Equation (1), and Hirsch *et al*. [25, 26] presented a filter flow approximation model. The estimation of kernels on edge-less regions is addressed by either penalizing the similarity of nearby kernels in the cost function [25] or interpolating the local kernels using nearby kernels [44]. Other works have explored the use of depth estimation for layer-wise deblurring, see *e.g*. [48].

Supervised learning for blind motion deblurring. For uniform motion blur of static scene, existing deep learning methods trained the network over many pairs of blurred/truth image. They either explicitly (*e.g*. [2–4]) or implicitly (*e.g*. [49]) estimate the motion-blur kernel.

For non-uniform motion blurring of static scene, there are both physics-based and physics-free methods. Sun *et al*. [6] trained a CNN to estimate local kernels, which were used to form a dense motion field via MRF. Gong *et al*. [7] predicted a flow field from a trained NN to define the blurring process. Tran *et al*. [18] proposed to learn a family of blurring operators in an encoded blur kernel space.

The physics-free methods directly train a DNN to map blurred images to clear images. The main differences among them lie in the design of the DNN. Aljadaany *et al*. [8] unfolded a Douglas-Rachford iteration. Different coarse-to-fine schemes are proposed in [13, 50, 51]. Kupyn *et al*. [52] used a GAN-based model, and Park *et al*. [9] leveraged multi-temporal training schemes. Depth information was exploited in [53]. However, training data is critical to the performance of supervised methods, which do not generalize well to images with blur effects that were not seen in the training samples.

Deblurring dynamic scenes is different from deblurring static scenes. Its focus is on separating moving objects and background. Different deep learning methods introduced different modules, *e.g*. attention modules [10–12, 54], deformable modules [12], nested-skip connection [55] pixel-adaptive RNN [56], and learning degradation model [19].

Semi-supervised deep learning for blind uniform motion deblurring. Lu *et al*. [21] trained the GAN with unpaired datasets for domain-specific deblurring, which might suffer from domain shift problems for wider adoption. Liu *et al*. [20] proposed another uniform deblurring NN with the input of multiple frames, generated by adjacent frames of video aligned by optical flow.

Dataset-free deep learning for blind uniform deblurring. Ren *et al*. [22] proposed a TV-based MAP estimator for blind uniform deblurring. In [22], two NNs are used to predict the latent image and blur kernel, with the prediction of the latent image regularized by TV. Chen *et al*. [24] proposed a DNN-based approach to blind uniform deblurring via an ensemble learning scheme. Li *et al*. [23] proposed a method that uses two NNs to represent the latent image and blur kernel, and learns the NN weights using Bayesian inference implemented via Monte-Carlo sampling.

3. Integrating DNN and MCEM

This paper proposes a dataset-free deep learning solution for removing general motion blur from static scene, including both uniform blur and non-uniform blur. However, since this is a dataset-free learning method, introducing certain prior on the blurring effect becomes necessary to address the training issue in the absence of truth images. For general motion blur, each pixel of the blurred image is a weighted average of its neighboring pixels, with the weights varying for different pixels. In this paper, we utilize the SVOLA

model (1) to characterize motion blur in static scenes, which encompasses uniform motion blur (2) as a specific instance.

DNN-based re-parametrization of the image $\boldsymbol{f}$ and kernel set $\boldsymbol{K}$. The CNN has an implicit prior on the images it represents, called DIP [57]. This prior favors regular image structures over random patterns when training a denoising CNN with only a noisy image. By reparametrizing an image using a CNN, we implicitly impose DIP on the latent image, which can help alleviate overfitting in a dataset-free setting. To represent the latent image, we adopt a 5-level U-Net CNN, as in [22]:

$$\boldsymbol{f} = T_f(\boldsymbol{\theta}_f, \boldsymbol{z}),$$

where $\boldsymbol{\theta}_f$ denotes NN weights and $\boldsymbol{z}$ denotes an initial seed.

To handle non-uniform motion blur, it is necessary to estimate many kernels, each of which is associated with a small image region. However, in the absence of truth images during training, the kernels can only be estimated in edge-rich regions and not in edge-free regions. To estimate kernels over all image regions, a prior on the kernel set must be introduced to share information among different kernels such that the information on edge-rich regions can be propagated to edge-free regions. In this paper, we propose a specific network structure with a specific prior for estimating the kernel set:

$$\boldsymbol{K} = T_K(\boldsymbol{\theta}_K, \tilde{\boldsymbol{z}}),$$

where $\boldsymbol{\theta}_K$ denotes NN weights and $\tilde{\boldsymbol{z}}$ is some initial seed.

Note that each kernel (point spread function) is associated with both the 3D camera motion and the scene depth. All kernels share the same 3D camera motion, meaning that there exists a lower-dimensional manifold for the kernels [44,58]. Two commonly used priors for the blur kernel are listed below.

1. *Implicit prior on the set of kernels*: There is certain implicit prior existing in the set of the kernels $\{\boldsymbol{k}_i\}$, as they are corresponding to the same 3D camera motion.
2. *Physical constraints*: Each element of the kernel $\boldsymbol{k}_i$ is non-negative, and the sum of all elements equals one.

To utilize these implicit priors, we designed a kernel network with shared components in the first stage for feature extraction. We used a U-Net with an encoder-decoder structure, whose output is used as the sole input in the second stage for predicting the kernels associated with different regions. In the second stage, a multi-head neural network is implemented, where each kernel has its own layers but shares the same input. The network outputs different kernels while maintaining certain correlations. We impose physical constraints on the kernels $\boldsymbol{k}_i$ by adding a softmax layer in the multi-tail output. Please refer to Figure 2 for the architecture and Figure 3 for an illustration of the correlation among the set of kernels.

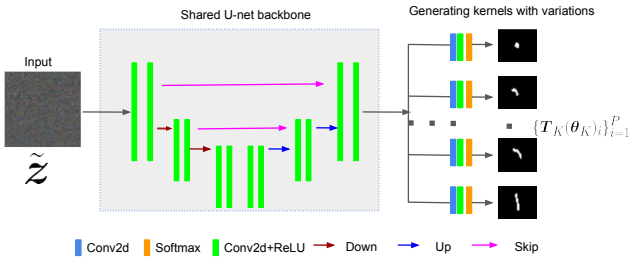


Figure 2. The structure of the U-Net $T_{\boldsymbol{K}}$ with shared component.

Outline of MCEM-based algorithm. As a reminder, we re-parameterize the two unknowns using two networks: $\boldsymbol{f} := T_f(\boldsymbol{\theta}_f, \boldsymbol{z})$ and the kernel set $\boldsymbol{K} := T_K(\boldsymbol{\theta}_K, \tilde{\boldsymbol{z}})$. In other words, the estimation of the image and kernel set is equivalent to estimating the NN weights $(\boldsymbol{\theta}_f, \boldsymbol{\theta}_K)$. Our goal is to train the two NNs to maximize the likelihood, motivated by the maximum likelihood (ML) estimator that estimates the image and kernel set $(\boldsymbol{f}, \boldsymbol{K})$:

$$\max_{\boldsymbol{\theta}_f, \boldsymbol{\theta}_K} \log p(\boldsymbol{g}|\boldsymbol{\theta}_f, \boldsymbol{\theta}_K). \tag{4}$$

For simplicity of notation, we omit the use of initials for randomly generated vectors $\boldsymbol{z}$ and $\tilde{\boldsymbol{z}}$ since they do not require estimation. Assuming Gaussian white noise $\boldsymbol{n} \sim \mathcal{N}(0, \sigma_g^2)$, the ML estimator for the image and kernel set $(\boldsymbol{f}, \boldsymbol{K})$ can be expressed as follows:

$$\min_{\boldsymbol{\theta}_f, \boldsymbol{\theta}_K} \frac{1}{2\sigma_g^2} \left\| F(T_f(\boldsymbol{\theta}_f), T_K(\boldsymbol{\theta}_K)) - \boldsymbol{g} \right\|_2^2. \tag{5}$$

However, the NNs trained using the loss function derived from the ML estimator may not effectively address the issue of over-fitting, as they tend to bias the solution towards the no-blur case $(\boldsymbol{g}, \boldsymbol{k}_i = \delta_i)$. Refer to Figure 3 (b) for an illustration of this bias.

Instead of replacing the ML estimator with the MAP estimator to address overfitting, we propose a self-training process derived from the EM algorithm. Instead of using the popular VEM method in existing works, we consider the MCEM algorithm [59, 60] for network training. The main difference between VEM and MCEM is that, in MCEM, the expectation in the E-step is numerically calculated through Monte Carlo simulation. The proposed MCEM algorithm is set up as follows.

1. Observation data: the blurred image $\boldsymbol{g}$
2. Latent variable: the weights $\boldsymbol{\theta}_f$ of image-relating network $T(\boldsymbol{\theta}_f, \boldsymbol{z})$
3. Parameters: the weights $\boldsymbol{\theta}_K$ of kernel-relating network $T(\boldsymbol{\theta}_K, \tilde{\boldsymbol{z}})$.

The EM algorithm estimates the parameters by iteratively maximizing the log-likelihood:

$$\begin{aligned} \boldsymbol{\theta}_{\boldsymbol{K}}^* &= \operatorname{argmax} \log p(\boldsymbol{g}|\boldsymbol{\theta}_{\boldsymbol{f}}, \boldsymbol{\theta}_{\boldsymbol{K}}) \\ &= \operatorname{argmax} \log \int p(\boldsymbol{g}|\boldsymbol{\theta}_{\boldsymbol{f}}; \boldsymbol{\theta}_{\boldsymbol{K}}) p(\boldsymbol{\theta}_f) d\boldsymbol{\theta}_{\boldsymbol{f}}. \end{aligned} \tag{6}$$

(a) (b) (c) (d)

Figure 3. Illustration of different algorithms for the DNN-based ML estimator with the loss (5). (a) Blurry image; (b) The image and kernel set estimated by gradient descent; (c) The image and kernel set estimated by the proposed MCEM algorithm; (d) The ground truth.

There are two steps in each iteration:

1. *E-step*: Calculating the expectation of logarithm likelihood with respective to $p(\boldsymbol{\theta}_f|\boldsymbol{g};\boldsymbol{\theta}_K^{t-1})$:
$$Q(\boldsymbol{\theta}_K|\boldsymbol{\theta}_K^{t-1}) = \mathbb{E}_{\boldsymbol{\theta_f}\sim p(\boldsymbol{\theta_f}|\boldsymbol{g};\boldsymbol{\theta}_K^{t-1})}\left[\log p(\boldsymbol{g}|\boldsymbol{\theta}_f;\boldsymbol{\theta}_K)\right]; \tag{7}$$
2. *M-step*: Maximize the expectation of the likelihood:
$$\boldsymbol{\theta}_K^t = \underset{\boldsymbol{\theta}_K}{\operatorname{argmax}}\, Q(\boldsymbol{\theta}_K|\boldsymbol{\theta}_K^{t-1}). \tag{8}$$

The computation of the expectation in the E-step is generally intractable for $p(\boldsymbol{\theta_f}|\boldsymbol{g};\boldsymbol{\theta}_K^{t-1})$. Instead of attempting to calculate the exact expectation, the MCEM method approximates the integral through Monte Carlo simulation:

$$Q(\boldsymbol{\theta}_K|\boldsymbol{\theta}_K^{t-1}) \approx \frac{1}{n_s}\sum_{i=1}^{n_s}\log p(\boldsymbol{g}|\boldsymbol{\theta}_f^i;\boldsymbol{\theta}_K), \boldsymbol{\theta}_f^i \sim p(\boldsymbol{\theta}_f|\boldsymbol{g};\boldsymbol{\theta}_K^{t-1}).$$

Once the NN weights $\boldsymbol{\theta}_f$ and $\boldsymbol{\theta}_K$ are estimated using the MCEM method, the estimated weights can be used to call the NNs for predicting the latent image.

Remark 1 (MCEM vs. VEM)**.** The main difference between VEM and MCEM lies in how the expectation is calculated in the E-step. In VEM, the expectation is approximated by replacing the distribution with a tractable one. In MCEM, the expectation is approximated by calculating the integral using Monte Carlo simulation. The motivation for using MCEM is its computational efficiency over VEM, particularly in the case of estimating DNN weights.

4. MCEM algorithm for dataset-free training

4.1. E-step and M-step

The MCEM algorithm consists of the E-step (7) and the M-step (8). In the E-step of MCEM, one approximates $Q(\boldsymbol{\theta_K}|\boldsymbol{\theta_K}^{t-1})$ using Monte Carlo simulation. This requires effectively sampling the posterior distribution $p(\boldsymbol{\theta_f}|\boldsymbol{g};\boldsymbol{\theta}_K^{t-1})$ and using the samples to approximate the integral through Monte Carlo simulation.

By Bayes rule, we have

$$p(\boldsymbol{\theta}_f|\boldsymbol{g};\boldsymbol{\theta}_K^{t-1}) \propto p(\boldsymbol{g}|\boldsymbol{\theta}_f;\boldsymbol{\theta}_K^{t-1})p(\boldsymbol{\theta}_f).$$

By assuming the prior distribution $p(\boldsymbol{\theta}_f)$ is an uniform distribution over a sufficiently large cube. We have then

$$p(\boldsymbol{\theta}_f|\boldsymbol{g};\boldsymbol{\theta}_K^{t-1}) \propto \exp\left(-\frac{\left\|F(T_f(\boldsymbol{\theta}_f), T_K(\boldsymbol{\theta}_K^{t-1})) - \boldsymbol{g}\right\|_2^2}{2\sigma_g^2}\right).$$

Motivated by the effectiveness of Langevin dynamics (LD) as a Monte Carlo sampler of NN weights in Bayesian deep learning [61], we propose to sample $p(\boldsymbol{\theta}_f|\boldsymbol{g};\boldsymbol{\theta}_K^{t-1})$ using LD. Interested readers can find a detailed introduction to LD in the supplementary file.

LD samples the distribution $p(\boldsymbol{\theta}_f|\boldsymbol{g};\boldsymbol{\theta}_K^{t-1})$ by the so-called *stochastic gradient Langevin dynamics* (SGLD): For $i = 1, 2, \ldots, n_s$

$$\boldsymbol{\theta}_f^i = \boldsymbol{\theta}_f^{i-1} + \alpha\nabla_{\boldsymbol{\theta}_f}\log p(\boldsymbol{\theta}_f^{i-1}|\boldsymbol{g};\boldsymbol{\theta}_K^{t-1}) + \sqrt{2\alpha}\boldsymbol{w}, \tag{9}$$

where $\boldsymbol{w} \sim \mathcal{N}(0, I)$. The hyperparameter α is the step size that satisfies the Robbins-Monro condition. The iteration scheme (9) enables us to quickly generate approximate samples from the posterior distribution $p(\boldsymbol{\theta}_f|\boldsymbol{g};\boldsymbol{\theta}_K^{t-1})$, which is used to approximate the expectation in the E-step:

$$Q(\boldsymbol{\theta}_K|\boldsymbol{\theta}_K^{t-1}) \approx \frac{1}{n_s}\sum_{i=1}^{n_s}\log p(\boldsymbol{g}|\boldsymbol{\theta}_f^i;\boldsymbol{\theta}_K), \tag{10}$$

where $\boldsymbol{\theta}_f^i$ are generated from (9). The M-step updates the estimates of the kernel set by solving the problem (10), which is equivalent to train the NN $T_K(\boldsymbol{\theta}_K)$ by minimizing corresponding loss function.

4.2. Warm-up trick

Good initialization of the kernel can improve the stability and computational efficiency of blind deblurring. Therefore, a warm-up strategy is implemented in the proposed training scheme. First, we train the NNs assuming all kernels in $\boldsymbol{K}$ are the same, and then copy the trained weights of the last Conv layer to initialize the weights of each kernel. In other words, at the beginning, the network is trained using a degenerate version of SVOLA which reads

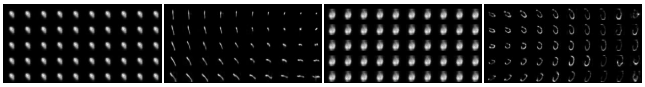
$$\min_{\boldsymbol{\theta_f},\boldsymbol{\theta_K}} \mathcal{L}_{\text{warm\_up}} := \|\boldsymbol{g} - [T_{\boldsymbol{K}}(\boldsymbol{\theta_K})]_1 \otimes T_{\boldsymbol{f}}(\boldsymbol{\theta_f})\|_2^2, \tag{11}$$

where $[T_{\boldsymbol{K}}(\boldsymbol{\theta_K})]_1$ denotes the first branch of the kernel set. After training the NN for a certain number of steps, we initialize the corresponding NN $\{T_{\boldsymbol{K}}(\widetilde{\boldsymbol{\theta}}_{\boldsymbol{K}})_i\}_{i=1}^P$ with the weights of the last convolution layer corresponding to

$[T_{\boldsymbol{K}}(\boldsymbol{\theta_K})]_1$. Then, we train the network using the full kernel set from the network T_K, and we optimize

$$\min_{\boldsymbol{\theta_f},\boldsymbol{\theta_K}} \mathcal{L} := \|\boldsymbol{g} - F(T_{\boldsymbol{f}}(\boldsymbol{\theta_f}), T_{\boldsymbol{K}}(\boldsymbol{\theta_K}))\|_2^2. \quad (12)$$

See Alg. 1 for the implementation and Figure 4 for visual comparison of two examples of the kernel estimate at the stage of warm-up training and at the last stage.



(a) Example 1 (b) Example 2

Figure 4. Kernels after the warm-up (left) and convergence (right).

Algorithm 1 Dataset-free non-uniform motion deblurring.

Input: Input image $\boldsymbol{g}$; No. of warm-up iterations N; No. of total iterations T; Grid size P; No. of samples n_s and No. of iteration for inner M-step n_ℓ.

Output: Estimated image $\boldsymbol{f}$ and the kernel set $\{\boldsymbol{k}_i\}_{i=1}^P$.

%% Warm-up training
for $t = 1 : N$ **do**
  Set $\boldsymbol{\theta}_f^0 = \boldsymbol{\theta}_f^{t-1}$ and produce n_s samples by
  $\boldsymbol{\theta}_f^i = \boldsymbol{\theta}_f^{i-1} - \alpha\nabla_{\boldsymbol{\theta}_f}\mathcal{L}_{\text{warm\_up}} + \sqrt{2\alpha}\boldsymbol{w}, i = 1, \ldots, n_s$
  Optimize $\frac{1}{n_s}\sum_{i=1}^{n_s}\|\boldsymbol{g} - [T_{\boldsymbol{K}}(\boldsymbol{\theta_K})]_1 \otimes T_{\boldsymbol{f}}(\boldsymbol{\theta}_{\boldsymbol{f}}^i)\|_2^2$ using n_ℓ Adam steps and obtain the solution $\boldsymbol{\theta}_{\boldsymbol{K}}^t$
To initialize $T_{\boldsymbol{K}}(\widetilde{\boldsymbol{\theta}}_{\boldsymbol{K}})$, use $T\boldsymbol{K}(\boldsymbol{\theta}\boldsymbol{K}^N)$ and replicate the last convolutional layer for predicting kernels.
%% Training
for $t = N+1 : T$ **do**
  Set $\boldsymbol{\theta}_f^0 = \boldsymbol{\theta}_f^{t-1}$ and produce n_s samples by
  $\boldsymbol{\theta}_f^i = \boldsymbol{\theta}_f^{i-1} - \alpha\nabla_{\boldsymbol{\theta}_f}\mathcal{L} + \sqrt{2\alpha}\boldsymbol{w}, i = 1, \ldots, n_s$
  Optimize $\frac{1}{n_s}\sum_{i=1}^{n_s}\|\boldsymbol{g} - F(T_{\boldsymbol{f}}(\boldsymbol{\theta}_{\boldsymbol{f}}^i), T_{\boldsymbol{K}}(\boldsymbol{\theta_K}))\|_2^2$ using n_ℓ Adam steps and obtain the solution $\boldsymbol{\theta}_{\boldsymbol{K}}^t$
$\boldsymbol{f} = T_{\boldsymbol{f}}(\boldsymbol{\theta}_{\boldsymbol{f}}^T); \{\boldsymbol{k}_i\}_{i=1}^P = \{T_{\boldsymbol{K}}(\widetilde{\boldsymbol{\theta}}_{\boldsymbol{K}}^T)_i\}_{i=1}^P$

5. Experiments

Parameter settings. The image NN $T_{\boldsymbol{f}}(\boldsymbol{\theta_f})$ is implemented as 5-level U-Net with channel size 64. The kernel NN $T_{\boldsymbol{K}}(\boldsymbol{\theta_K})$ is implemented as U-Net with 4 levels whose channel size is $[32, 32, 64, 64]$. The learning rate is set to be 0.01 for $T_{\boldsymbol{f}}$ and 0.0001 for M-step when optimizing $T_{\boldsymbol{K}}$. They are halved after 2000 and 3000 iterations. We train our framework for $T = 5000$ steps with warming up $N = 500$ steps. The grid size P is set to 5×10. The code is publicly accessed at https://github.com/Chilie/Deblur_MCEM.

Algorithm 1 consists of two inner loops: one is an iterative scheme (9) for generating samples to approximate the function Q, and the other is an iterative scheme for minimizing the loss (10) using gradient descent. An empirical

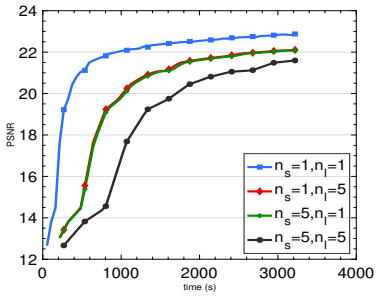


Figure 5. The PSNR gain over elapsed time for different (n_s, n_ℓ).

study on the performance of Algorithm 1 on a sample image with different configurations of (n_s, n_ℓ) is shown in Figure 5. The results indicate that the values of (n_s, n_ℓ) have a greater impact on computational efficiency than on performance. Therefore, for computational efficiency, we set $(n_s, n_\ell) = (1, 1)$ in our experiments.

Network architecture for the image. The same U-net as [22, 57] is adopted for image network $T_{\boldsymbol{f}}$. It contains five downsampling modules and upsampling modules with skip connections. Apart from the Down/Up sampling layer, each module also comprises the Conv $\rightarrow$ BN $\rightarrow$ ReLU layers. As image deblurring is sensitive to noise, the Decoder part is then trained with dropout to regularize the prediction. Finally, a 1×1 convolution layer is used to generate the image. To keep the values of the image in the range $[0, 1]$, we concatenate a Sigmoid layer to the output layer.

5.1. Evaluation on non-uniform motion deblurring

This study aims to compare our proposed blind deblurring method with existing methods for deblurring static scenes[1]. In the comparison table, the best result is highlighted in blue, and the second-best result is underlined. Note that the scores are calculated with alignment to account for possible shift ambiguity in the results.

Benchmark dataset from Köhler *et al.* [62]. Köhler *et al.*'s dataset is a real non-uniform dataset that records the motion-blurred images caused by 6-dimensional camera motions. It contains 48 images captured in 4 scenes, each with 12 motion trajectories. These trajectories exhibit diverse levels of non-uniformity ranging from mild to severe. Table 1 presents a performance comparison between our proposed method and other methods, including both non-learning and supervised deep-learning methods, on this dataset. On average, our method outperforms the second-best method by 0.37 dB in PSNR and 0.013 in MSSIM.

Benchmark dataset from Lai *et al.* [66]. The dataset contains 100 images that were obtained by recording camera motions using inertial sensors in a cellphone. The spatially-varying blur kernels were constructed from these motion

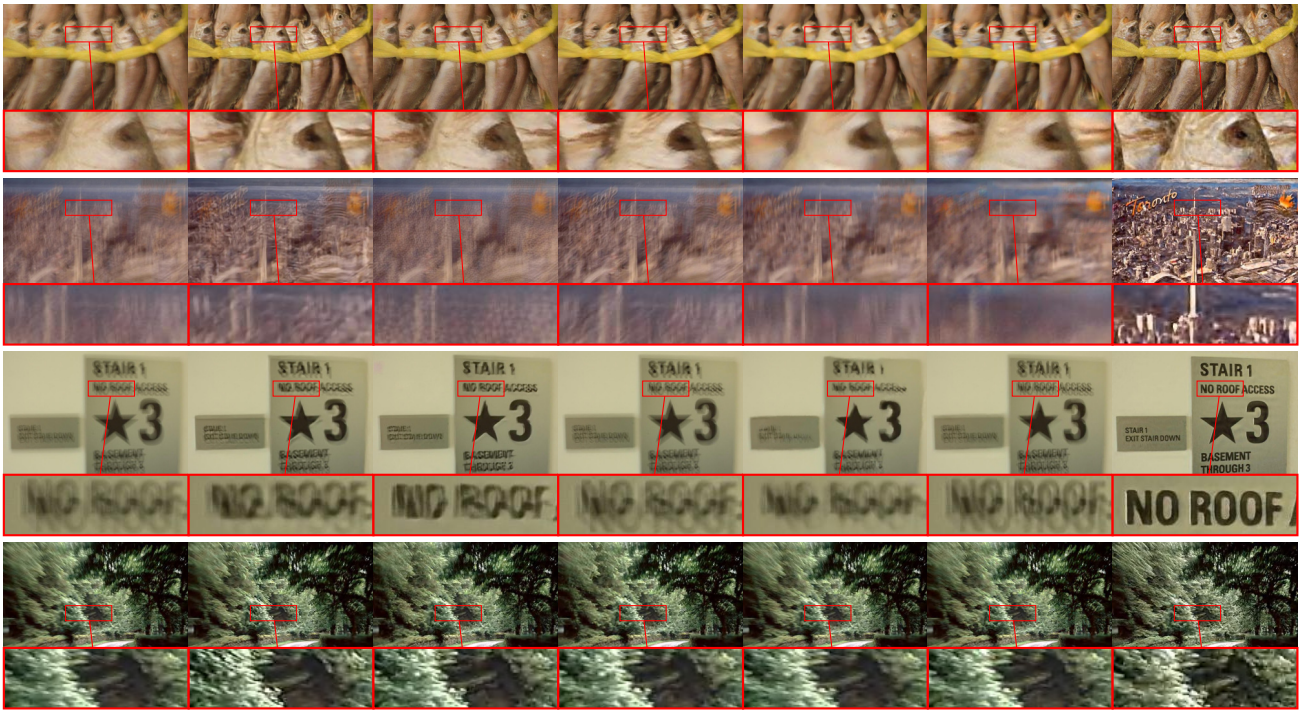
[1]The method [35] used in this study is not its uniform version, but the extension from the authors for handling non-uniform blurred images.

Table 1. Average PSNR/MSSIM comparison on the non-uniform dataset of Köhler *et al*. [62]

No.	Non-learning methods			Supervised learning methods					Self-supervised	
	Xu *et al*. 2013' [35]	Whyte *et al*. 2014' [63]	Vasu *et al*. 2017' [64]	Tao *et al*. 2018' [50]	Kupyn *et al*. 2019' [52]	Zamir *et al*. 2021' [65]	Cho *et al*. 2021' [13]	Li *et al*. 2022' [19]	Liu *et al*. 2020' [20]	Ours
1	29.19/0.806	29.77/0.850	32.44/0.940	29.14/0.850	28.99/0.845	29.86/0.865	27.66/0.812	29.82/0.866	27.21/0.808	32.41/0.942
2	24.43/0.757	24.27/0.809	26.52/0.927	23.10/0.798	23.78/0.805	22.57/0.782	21.69/0.741	22.56/0.784	21.19/0.736	26.84/0.932
3	29.97/0.854	30.73/0.887	32.60/0.936	29.96/0.887	30.00/0.874	28.04/0.859	28.09/0.846	30.21/0.889	28.20/0.853	33.18/0.951
4	25.76/0.766	26.60/0.818	27.99/0.906	25.22/0.820	25.09/0.798	24.78/0.803	23.91/0.765	24.95/0.807	23.49/0.757	28.60/0.936
Avg.	27.34/0.796	27.84/0.841	29.89/0.927	26.85/0.839	26.97/0.830	26.32/0.827	25.34/0.791	26.89/0.837	25.02/0.789	30.26/0.940

Table 2. Average PSNR/SSIM comparison on the non-uniform dataset of Lai *et al*. [66]

	Non-learning methods			Supervised learning methods					Self-supervised	
	Xu *et al*. 2013' [35]	Whyte *et al*. 2014' [63]	Vasu *et al*. 2017' [64]	Tao *et al*. 2018' [50]	Kupyn *et al*. 2019' [52]	Zamir *et al*. 2021' [65]	Cho *et al*. 2021' [13]	Li *et al*. 2022' [19]	Liu *et al*. 2020' [20]	Ours
Manmade	17.90/0.497	17.33/0.433	17.93/0.474	18.45/0.488	18.73/0.512	17.42/0.435	16.78/0.379	17.28/0.414	17.39/0.452	19.17/0.572
Natural	21.99/0.607	21.04/0.543	21.94/0.598	22.28/0.610	22.24/0.601	20.76/0.530	19.88/0.474	20.59/0.513	20.90/0.518	22.70/0.641
People	25.42/0.801	23.92/0.746	25.63/0.803	26.87/0.834	26.71/0.828	23.95/0.770	23.64/0.754	24.23/0.777	24.76/0.770	26.90/0.825
Saturated	18.39/0.644	17.33/0.606	17.57/0.612	20.10/0.699	17.91/0.618	16.73/0.561	16.58/0.539	16.67/0.551	18.52/0.529	21.46/0.754
Text	18.97/0.749	13.22/0.420	19.19/0.765	18.66/0.756	19.11/0.781	15.63/0.608	17.17/0.668	17.45/0.696	17.42/0.660	21.91/0.872
Average	20.53/0.660	18.57/0.550	20.45/0.650	21.27/0.677	20.94/0.668	18.90/0.581	18.81/0.563	19.25/0.590	19.80/0.581	22.42/0.738



Input | Tao *et al*. [50] | Kupyn *et al*. [52] | Park *et al*. [9] | Zamir *et al*. [65] | Li *et al*. [19] | Ours

Figure 6. Visual comparison of the results for samples images from real dataset of Lai *et al*. [66] and Sun *et al*. [6]. More visual comparison can be found in supplementary file.

Table 3. Ablation study on Lai *et al*.'s dataset

MLE (5)	MAP ((5)+TV)	EM w/o warm-up	Proposed MCEM
19.49/0.609	19.97/0.643	21.14/0.669	22.42/0.738

trajectories, assuming constant scene depth. The blurred images were generated by convolving the latent images with these kernels and adding 1% Gaussian noise. The images were divided into 5 categories, each containing 20 images,

Table 4. Average PSNR/SSIM for uniform blind deblurring on Lai *et al.*'s dataset [66].

	Non-learning methods			Supervised learning methods						Self-supervised	
Metric	Xu & Jia 2010' [36]	Yan *et al.* 2017' [42]	Yang & Ji 2019' [32]	Tao *et al.* 2018' [50]	Kupyn *et al.* 2019 [52]	Kaufman *et al.* 2020' [5]	Zamir *et al.* 2021' [65]	Cho *et al.* 2021' [13]	Li *et al.* 2022' [19]	Ren *et al.* 2020' [22]	Ours
PSNR	20.93	21.12	<u>21.79</u>	16.72	17.02	20.18	16.15	16.36	16.43	21.11	23.62
SSIM	0.654	0.673	<u>0.704</u>	0.471	0.49	0.643	0.454	0.461	0.469	0.671	0.766

with complex scenes and severe non-uniform blur effects.

Table 2 presents the comparison of our proposed method with other non-learning and supervised learning methods on this dataset. The results show that our method outperforms the second-best performer by an average of <u>1.15</u> dB in PSNR and <u>0.061</u> in SSIM. These results demonstrate that our method provides a significant performance gain over existing methods for the challenging Lai *et al.*'s dataset.

Visual comparison on real-world images. As ground truth are not available for real-world images, we have only included visual comparisons of a few deblurred results from different methods. The sample images are taken from both Lai *et al.* [66]'s dataset and Sun *et al.* [6]'s dataset. Visual comparisons of more examples can be found in the supplementary file. As shown in Figure 6 and the supplementary file, the proposed method yields results of the best visual quality, with fewer artifacts and sharper details.

5.2. Evaluation on uniform motion deblurring

The proposed method is designed to remove general motion blur caused by camera shake from a single image. Thus, it is applicable to both non-uniform motion blur and uniform motion blur of static scenes. Uniform motion blur occurs when there is only in-plane camera translation and constant scene depth. By reducing the SVOLA model to the case with a single kernel $P = 1$, we obtain a convolution model (2) for modeling uniform motion blur.

To evaluate the performance of the proposed general blind deblurring method on uniform motion deblurring, we conducted an experiment on one popular benchmark dataset, Lai *et al.*'s dataset [66]. Table 4 presents the results of representative non-learning methods, recent supervised-learning methods and dataset-free method. Our proposed method performs very competitively against these blind uniform deblurring methods, as shown in Table 4. More related experiments can be found in the supplementary file.

5.3. Ablation study

MCEM for ML estimator of $\boldsymbol{K}$ vs ML estimator of $(\boldsymbol{f}, \boldsymbol{K})$. The proposed MCEM method is an EM approach designed to find the ML estimation of $\boldsymbol{K}$. It is compared to the results obtained by training DNNs using the loss function (5), referred to as "MLE", which aims to find the MLE of both $\boldsymbol{f}$ and $\boldsymbol{K}$. The performance gain achieved by the proposed MCEM algorithm is approximately 2.9 dB, demonstrating its effectiveness for training. While the TV regularization on the image ($\lambda = 5e-2$) does improve the "MLE" performance, its effectiveness is much lower than that of the MCEM method.

With vs without warm-up training. The warm-up training strategy not only speeds up the training process, but also provides a good initialization for the NNs. As shown in Table 3, compared to the model trained without warm-up, the warm-up training strategy results in a performance gain of about 1.3 dB, indicating the significant benefit of the warm-up training strategy.

5.4. Limitation

The proposed method is designed for removing general motion blur caused by camera shake from a single image. Recovering dynamic scenes with moving objects is not within the scope of the proposed method, as the SVOLA model cannot capture the "blending" effect around the boundary of moving objects. A limited experiment is conducted by applying the proposed method on both static scenes and dynamic scenes from GOPRO [17], a dataset for dynamic scene deblurring. Our method works well on those images dominated by static scenes, but not so on those images of dynamic scenes. See the supplementary file for more related experimental results.

6. Conclusion

This paper proposes a self-supervised deep learning method that can remove general motion blur (both uniform and non-uniform) from a single image, without requiring a dataset. The main idea is to train NNs using a scheme derived from the MCEM algorithm for maximum likelihood estimation. Our method outperforms existing non-learning and deep learning methods, including both supervised and self-supervised approaches, as demonstrated by extensive experiments on standard benchmark datasets. In the future, we plan to investigate the extension of our proposed method to recover motion-blurred images of dynamic scenes

Acknowledgment

The authors would like to thank the support by Singapore MOE Academic Research Fund (AcRF) Tier 1 with WBS number A-8000981-00-00.

References

[1] Wangmeng Zuo, Dongwei Ren, David Zhang, Shuhang Gu, and Lei Zhang. Learning iteration-wise generalized shrinkage–thresholding operators for blind deconvolution. *IEEE Trans. Image Process.*, 25(4):1751–1764, 2016. 1

[2] Xiangyu Xu, Jinshan Pan, Yu-Jin Zhang, and Ming-Hsuan Yang. Motion blur kernel estimation via deep learning. *IEEE Trans. Image Process.*, 27(1):194–205, 2017. 1, 3

[3] Christian J Schuler, Michael Hirsch, Stefan Harmeling, and Bernhard Schölkopf. Learning to deblur. *IEEE Trans. Pattern Anal. Mach. Intell.*, 38(7):1439–1451, 2015. 1, 3

[4] Lerenhan Li, Jinshan Pan, Wei-Sheng Lai, Changxin Gao, Nong Sang, and Ming-Hsuan Yang. Learning a discriminative prior for blind image deblurring. In *CVPR*, pages 6616–6625, 2018. 1, 3

[5] Adam Kaufman and Raanan Fattal. Deblurring using analysis-synthesis networks pair. In *CVPR*, June 2020. 1, 8

[6] Jian Sun, Wenfei Cao, Zongben Xu, and Jean Ponce. Learning a convolutional neural network for non-uniform motion blur removal. In *CVPR*, pages 769–777, 2015. 1, 3, 7, 8

[7] Dong Gong, Jie Yang, Lingqiao Liu, Yanning Zhang, Ian Reid, Chunhua Shen, Anton Van Den Hengel, and Qinfeng Shi. From motion blur to motion flow: a deep learning solution for removing heterogeneous motion blur. In *CVPR*, pages 2319–2328, 2017. 1, 3

[8] Raied Aljadaany, Dipan K Pal, and Marios Savvides. Douglas-Rachford networks: learning both the image prior and data fidelity terms for blind image deconvolution. In *CVPR*, pages 10235–10244, 2019. 1, 3

[9] Dongwon Park, Dong Un Kang, Jisoo Kim, and Se Young Chun. Multi-temporal recurrent neural networks for progressive non-uniform single image deblurring with incremental temporal training. In *ECCV*, 2020. 1, 3, 7

[10] Maitreya Suin, Kuldeep Purohit, and AN Rajagopalan. Spatially-attentive patch-hierarchical network for adaptive motion deblurring. In *CVPR*, pages 3606–3615, 2020. 1, 3

[11] Kaihao Zhang, Wenhan Luo, Yiran Zhong, Lin Ma, Bjorn Stenger, Wei Liu, and Hongdong Li. Deblurring by realistic blurring. In *CVPR*, pages 2737–2746, 2020. 1, 3

[12] Yuan Yuan, Wei Su, and Dandan Ma. Efficient dynamic scene deblurring using spatially variant deconvolution network with optical flow guided training. In *CVPR*, pages 3555–3564, 2020. 1, 3

[13] Sung-Jin Cho, Seo-Won Ji, Jun-Pyo Hong, Seung-Won Jung, and Sung-Jea Ko. Rethinking coarse-to-fine approach in single image deblurring. In *ICCV*, pages 4641–4650, 2021. 1, 3, 7, 8

[14] Seo-Won Ji, Jeongmin Lee, Seung-Wook Kim, Jun-Pyo Hong, Seung-Jin Baek, Seung-Won Jung, and Sung-Jea Ko. XYDeblur: divide and conquer for single image deblurring. In *CVPR*, pages 17421–17430, 2022. 1

[15] Jay Whang, Mauricio Delbracio, Hossein Talebi, Chitwan Saharia, Alexandros G Dimakis, and Peyman Milanfar. Deblurring via stochastic refinement. In *CVPR*, pages 16293–16303, 2022. 1

[16] Jaesung Rim, Haeyun Lee, Jucheol Won, and Sunghyun Cho. Real-world blur dataset for learning and benchmarking deblurring algorithms. In *ECCV*, pages 184–201. Springer, 2020. 1

[17] Seungjun Nah, Tae Hyun Kim, and Kyoung Mu Lee. Deep multi-scale convolutional neural network for dynamic scene deblurring. In *CVPR*, pages 3883–3891, 2017. 1, 8

[18] Phong Tran, Anh Tuan Tran, Quynh Phung, and Minh Hoai. Explore image deblurring via encoded blur kernel space. In *CVPR*, pages 11956–11965, 2021. 1, 3

[19] Dasong Li, Yi Zhang, Ka Chun Cheung, Xiaogang Wang, Hongwei Qin, and Hongsheng Li. Learning degradation representations for image deblurring. In *ECCV*, 2022. 1, 3, 7, 8

[20] Peidong Liu, Joel Janai, Marc Pollefeys, Torsten Sattler, and Andreas Geiger. Self-supervised linear motion deblurring. *IEEE Robot. Autom. Lett.*, 5(2):2475–2482, 2020. 2, 3, 7

[21] Boyu Lu, Jun-Cheng Chen, and Rama Chellappa. Unsupervised domain-specific deblurring via disentangled representations. In *CVPR*, pages 10225–10234, 2019. 2, 3

[22] Dongwei Ren, Kai Zhang, Qilong Wang, Qinghua Hu, and Wangmeng Zuo. Neural blind deconvolution using deep priors. In *CVPR*, pages 3341–3350, 2020. 2, 3, 4, 6, 8

[23] Ji Li, Yuesong Nan, and Hui Ji. Un-supervised learning for blind image deconvolution via Monte-Carlo sampling. *Inverse Probl*, 38(3):035012, 2022. 2, 3

[24] Mingqin Chen, Yuhui Quan, Yong Xu, and Hui Ji. Self-supervised blind image deconvolution via deep generative ensemble learning. *IEEE Trans. Circuits Syst. Video Technol.*, 2022. 2, 3

[25] Stefan Harmeling, Hirsch Michael, and Bernhard Schölkopf. Space-variant single-image blind deconvolution for removing camera shake. In *NIPS*, pages 829–837, 2010. 2, 3

[26] Michael Hirsch, Suvrit Sra, Bernhard Schölkopf, and Stefan Harmeling. Efficient filter flow for space-variant multiframe blind deconvolution. In *CVPR*, pages 607–614. IEEE, 2010. 2, 3

[27] Yeong Ho Ha and John A Pearce. A new window and comparison to standard windows. *IEEE Trans. Acoust., Speech, Signal Process.*, 37(2):298–301, 1989. 2

[28] Anat Levin, Yair Weiss, Fredo Durand, and William T Freeman. Understanding and evaluating blind deconvolution algorithms. In *CVPR*, pages 1964–1971, 2009. 2, 3

[29] Anat Levin, Yair Weiss, Fredo Durand, and William T Freeman. Efficient marginal likelihood optimization in blind deconvolution. In *CVPR*, pages 2657–2664. IEEE, 2011. 2, 3

[30] D. Wipf and H. Zhang. Revisiting bayesian blind deconvolution. *J. Mach. Learn. Res.*, 15:3775–3814, 2014. 2, 3

[31] S. D. Babacan, R. Molina, M. N. Do, and A. K. Katsaggelos. Bayesian blind deconvolution with general sparse image priors. In *ECCV*, pages 341–355, 2012. 2, 3

[32] Liuge Yang and Hui Ji. A variational EM framework with adaptive edge selection for blind motion deblurring. In *CVPR*, pages 10167–10176, 2019. 2, 3, 8

[33] Jinshan Pan, Zhe Hu, Zhixun Su, and Ming-Hsuan Yang. Deblurring text images via ℓ_0-regularized intensity and gradient prior. In *CVPR*, pages 2901–2908, 2014. 3

[34] Dilip Krishnan, Terence Tay, and Rob Fergus. Blind deconvolution using a normalized sparsity measure. In *CVPR*, pages 233–240, 2011. 3

[35] Li Xu, Shicheng Zheng, and Jiaya Jia. Unnatural ℓ_0 sparse representation for natural image deblurring. In *CVPR*, pages 1107–1114, 2013. 3, 6, 7

[36] Li Xu and Jiaya Jia. Two-phase kernel estimation for robust motion deblurring. In *ECCV*, pages 157–170, 2010. 3, 8

[37] Jian-Feng Cai, Hui Ji, Chaoqiang Liu, and Zuowei Shen. Blind motion deblurring from a single image using sparse approximation. In *CVPR*, pages 104–111, 2009. 3

[38] Jian-Feng Cai, Hui Ji, Chaoqiang Liu, and Zuowei Shen. Framelet-based blind motion deblurring from a single image. *IEEE Trans. Image Process.*, 21(2):562–572, 2011. 3

[39] Tomer Michaeli and Michal Irani. Blind deblurring using internal patch recurrence. In *ECCV*, pages 783–798, 2014. 3

[40] Libin Sun, Sunghyun Cho, Jue Wang, and James Hays. Edge-based blur kernel estimation using patch priors. In *ICCP*, pages 1–8, 2013. 3

[41] Jinshan Pan, Deqing Sun, Hanspeter Pfister, and Ming-Hsuan Yang. Blind image deblurring using dark channel prior. In *CVPR*, pages 1628–1636, 2016. 3

[42] Yanyang Yan, Wenqi Ren, Yuanfang Guo, Rui Wang, and Xiaochun Cao. Image deblurring via extreme channels prior. In *CVPR*, pages 4003–4011, 2017. 3, 8

[43] Rob Fergus, Barun Singh, Aaron Hertzmann, Sam T Roweis, and William T Freeman. Removing camera shake from a single photograph. *TOG*, 25(3):787–794, 2006. 3

[44] Hui Ji and Kang Wang. A two-stage approach to blind spatially-varying motion deblurring. In *CVPR*, pages 73–80. IEEE, 2012. 3, 4

[45] Yu-Wing Tai, Ping Tan, and Michael S Brown. Richardson-lucy deblurring for scenes under a projective motion path. *IEEE Trans. Pattern Anal. Mach. Intell.*, 33(8):1603–1618, 2011. 3

[46] Oliver Whyte, Josef Sivic, Andrew Zisserman, and Jean Ponce. Non-uniform deblurring for shaken images. *IJCV*, 98(2):168–186, 2012. 3

[47] Ankit Gupta, Neel Joshi, C Lawrence Zitnick, Michael Cohen, and Brian Curless. Single image deblurring using motion density functions. In *ECCV*, pages 171–184. Springer, 2010. 3

[48] Zhe Hu, Li Xu, and Ming-Hsuan Yang. Joint depth estimation and camera shake removal from single blurry image. In *CVPR*, pages 2893–2900, 2014. 3

[49] Li Xu, Jimmy SJ Ren, Ce Liu, and Jiaya Jia. Deep convolutional neural network for image deconvolution. In *NIPS*, pages 1790–1798, 2014. 3

[50] Xin Tao, Hongyun Gao, Xiaoyong Shen, Jue Wang, and Jiaya Jia. Scale-recurrent network for deep image deblurring. In *CVPR*, 2018. 3, 7, 8

[51] Hongguang Zhang, Yuchao Dai, Hongdong Li, and Piotr Koniusz. Deep stacked hierarchical multi-patch network for image deblurring. In *CVPR*, pages 5978–5986, 2019. 3

[52] Orest Kupyn, Tetiana Martyniuk, Junru Wu, and Zhangyang Wang. Deblurgan-v2: deblurring (orders-of-magnitude) faster and better. In *ICCV*, pages 8878–8887, 2019. 3, 7, 8

[53] Lerenhan Li, Jinshan Pan, Wei-Sheng Lai, Changxin Gao, Nong Sang, and Ming-Hsuan Yang. Dynamic scene deblurring by depth guided model. *IEEE Trans. Image Process.*, 29:5273–5288, 2020. 3

[54] Yong Xu, Ye Zhu, Yuhui Quan, and Hui Ji. Attentive deep network for blind motion deblurring on dynamic scenes. *Comput Vis Image Und*, 205:103169, 2021. 3

[55] Hongyun Gao, Xin Tao, Xiaoyong Shen, and Jiaya Jia. Dynamic scene deblurring with parameter selective sharing and nested skip connections. In *CVPR*, pages 3848–3856, 2019. 3

[56] Jiawei Zhang, Jinshan Pan, Jimmy Ren, Yibing Song, Linchao Bao, Rynson WH Lau, and Ming-Hsuan Yang. Dynamic scene deblurring using spatially variant recurrent neural networks. In *CVPR*, pages 2521–2529, 2018. 3

[57] Dmitry Ulyanov, Andrea Vedaldi, and Victor Lempitsky. Deep image prior. In *CVPR*, pages 9446–9454, 2018. 4, 6

[58] Neel Joshi, Sing Bing Kang, C Lawrence Zitnick, and Richard Szeliski. Image deblurring using inertial measurement sensors. *TOG*, 29(4):1–9, 2010. 4

[59] Christian P Robert, George Casella, and George Casella. *Monte Carlo statistical methods*, volume 2. Springer, 1999. 4

[60] Richard A Levine and George Casella. Implementations of the Monte Carlo EM algorithm. *J Comput Graph Stat*, 10(3):422–439, 2001. 4

[61] Max Welling and Yee W Teh. Bayesian learning via stochastic gradient Langevin dynamics. In *ICML*, pages 681–688, 2011. 5

[62] Rolf Köhler, Michael Hirsch, Betty Mohler, Bernhard Schölkopf, and Stefan Harmeling. Recording and playback of camera shake: benchmarking blind deconvolution with a real-world database. In *ECCV*, pages 27–40. Springer, 2012. 6, 7

[63] Oliver Whyte, Josef Sivic, and Andrew Zisserman. Deblurring shaken and partially saturated images. *IJCV*, 110(2):185–201, 2014. 7

[64] Subeesh Vasu and AN Rajagopalan. From local to global: edge profiles to camera motion in blurred images. In *CVPR*, pages 4447–4456, 2017. 7

[65] Syed Waqas Zamir, Aditya Arora, Salman Khan, Munawar Hayat, Fahad Shahbaz Khan, Ming-Hsuan Yang, and Ling Shao. Multi-stage progressive image restoration. In *CVPR*, pages 14821–14831, 2021. 7, 8

[66] Wei-Sheng Lai, Jia-Bin Huang, Zhe Hu, Narendra Ahuja, and Ming-Hsuan Yang. A comparative study for single image blind deblurring. In *CVPR*, pages 1701–1709, 2016. 6, 7, 8