RefDeblur: Blind Motion Deblurring with Self-Generated Reference Image

Anonymous authors
Paper under double-blind review

Abstract

The challenge of blind motion deblurring is often tackled via two distinct paradigms: kernel-based and kernel-free methods. Each deblurring method provides inherent strengths. Kernel-based methods facilitate generating texture-detailed sharp images by closely aligning with the blurring process. In contrast, kernel-free methods are more effective in handling complex blur patterns. Building upon these complementary benefits, we propose a hybrid framework that decomposes a non-uniform deblurring task into two simpler tasks: a uniform kernel estimation, managed by our kernel-based method, and error prediction, handled by our kernel-free method. Our kernel-based method serves to generate a reference image with realistic texture details while our kernel-free model refines the reference image by correcting residual errors with preserving texture details. To efficiently build our kernel-based model, we consider the logarithmic fourier space that facilitates estimating a blur kernel easier by simplifying the relationship between blur and sharp samples. Furthermore, the regime under using a texture-detailed reference image allows for reducing the size of our kernel-free model without compromising performance. As a result, the proposed method achieves remarkable performance on several datasets such as RealBlur, RSBlur and GoPro, and comparable performance to state-of-the-art methods with a 75% reduction in computational costs.

1 Introduction

Motion blur images often appear in low-light environments. They are caused by camera motion and object movement in the long exposure time. In low-light conditions, using a long exposure setting reduces noise but results in substantial blur. The goal of blind image deblurring is to restore sharp images from blur ones.

In earlier years, various deep methods have been devoted to investigating accurate blur kernel estimation (Schuler et al., 2015; Chakrabarti, 2016; Gong et al., 2017; Tran et al., 2021; Zhang et al., 2021; Carbajal et al., 2023). These kernel-based techniques prioritize estimating the blur kernels, which are then used in the deconvolution process (Fish et al., 1995; Krishnan & Fergus, 2009) to generate sharp images. Although they show promising results in certain cases, their performance is not acceptable for real-world scenarios. In fact, there are some drawbacks to kernel-based methods. First, they train with synthesized motion blur images that may not hold in practice. Second, estimating a blur kernel for every pixel is a huge ill-posed problem. Therefore, they may lead to unfavorable ringing artifacts due to inaccurate kernels, especially in noise and saturation pixels (Yuan et al., 2007; Whyte et al., 2014). Third, finding accurate per-pixel kernels may lead to increasing model size, and they mainly come with a time-consuming deconvolution procedure (Fish et al., 1995). In recent years, kernel-free methods have become more popular in motion deblurring tasks (Tao et al., 2018; Kupyn et al., 2018; 2019; Zamir et al., 2021; Cho et al., 2021; Mao et al., 2023; Nah et al., 2022; Tu et al., 2022; Zamir et al., 2022; Wang et al., 2022; Chen et al., 2022; Tsai et al., 2022; Kong et al., 2023; Li et al., 2023), exhibiting strong results without relying on blur kernels. However, they are primarily influenced by the network design (Ronneberger et al., 2015; He et al., 2016; Dosovitskiy et al., 2021; Tolstikhin et al., 2021) and capacity (He et al., 2016; Zagoruyko & Komodakis, 2016; Tan & Le, 2019).

While the kernel-based method can produce better texture-detailed sharp images over the kernel-free method by closely following the blurring process, the kernel-free method is beneficial for effectively managing complex

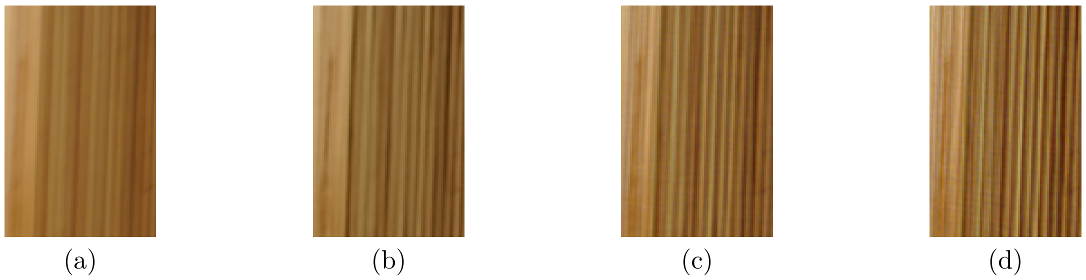


Figure 1: Visual comparison of the reference images: (a) Blur image, (b) NAFNet (Chen et al., 2022), (c) Our kernel-based model with $\lambda = 0.1$, and (d) Our kernel-based model with $\lambda = 0.01$.

blur patterns. Inspired by this, we propose a hybrid scheme that combines the strengths of kernel-based and kernel-free deblurring methods. Specifically, we decompose a non-uniform deblurring task into a uniform kernel estimation (e.g., managed by our kernel-based method) and error prediction tasks (e.g., managed by our kernel-free method). Our kernel-based model is responsible for estimating a blur kernel, which is used to generate a texture-detailed reference image. Then, our kernel-free model eliminates some errors in the self-generated reference images while preserving the remaining texture details. According to Reference-based Super-Resolution (RefSR) tasks (Zhang et al., 2019; Jiang et al., 2021), they suggest using an additional high-resolution reference image to benefit from meaningful details of the reference image. Namely, leveraging the additional self-generated reference image is a key strategy of our method to recover the realistic details of the deblurred image. In our experiments, we found that the rich-detailed reference image generated from our kernel-based method as shown in Fig. 1 (d) is preferred rather than the artifact-free reference image generated from the recent kernel-free methods as shown in Fig. 1 (b). This is because high texture details of the reference image can be contributed to restoring the realistic details for deblurring tasks.

To build an efficient kernel-based model for reference training, we pay attention to the Logarithmic Fourier Transform (LogFT) (Brigham & Morrow, 1967; Childers et al., 1977). Our intuition is that the logarithmic fourier space allows for simplifying the blur-sharp relation from complicated deconvolution to subtraction so that learning uniform kernels becomes easier that we can use a light-weight kernel-based model. Furthermore, equipped with a texture-detailed reference image, we can achieve a scaled-down version of our kernel-free model, whose performance is still comparable to that of state-of-the-art methods (Tu et al., 2022; Wang et al., 2022; Chen et al., 2022; Tsai et al., 2022) as shown in Table 1. Extensive experiments have been conducted to demonstrate the superiority of the proposed method under various datasets such as RealBlur (Rim et al., 2020), GoPro (Nah et al., 2017), and RSBlur (Rim et al., 2022) datasets.

2 Related Works

Kernel-based methods. Kernel-based methods (Schuler et al., 2015; Chakrabarti, 2016; Sun et al., 2015) estimate per-pixel kernels and subsequently apply existing non-blind deblurring methods (Fish et al., 1995; Krishnan & Fergus, 2009) to produce sharp images. To simplify blur kernel estimation, a motion flow estimation method (Gong et al., 2017) is explored to generate motion flow maps. Similarly, MotionETR (Zhang et al., 2021) is developed to estimate motion offsets, which are subsequently used to reconstruct sharp images. J-MKPD (Carbajal et al., 2023) proposes an efficient solution by introducing a set of pixel-shared motion bases along with pixel-wise mixing coefficients to accurately estimate per-pixel kernels. Most kernel-based methods struggle to use them in real-world scenarios. This is because they are mainly trained with synthetic data to learn blur kernels or motions. On the other hand, our kernel-based method not only learns blur kernels from realistic blur-sharp pair data (Rim et al., 2020; 2022), but also plays a role to provide a reference image to our kernel-free method for better motion deblurring.

Kernel-free methods. The kernel-free methods (Nah et al., 2017; Cho et al., 2021; Chen et al., 2022; Kong et al., 2023; Cui et al., 2024) are based on image-to-image translation tasks, mapping blur images to sharp ones in realistic blur datasets (Rim et al., 2020; 2022). Notably, earlier works (Tao et al., 2018; Nah et al., 2017) introduce a coarse-to-fine strategy that progressively reconstructs sharp images, starting from low-resolution to high-resolution images. In order to implement this strategy more efficiently, some

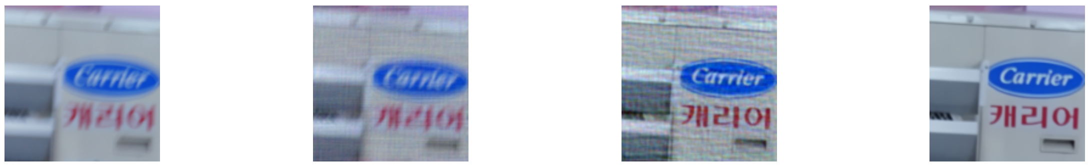


Figure 2: Visual results on the proposed method: (a) Blur image (input), (b) Estimated blur image by equation 6, (c) Estimated reference image by equation 5, and (d) Final deblurred image by equation 8

literature (Zamir et al., 2021; Cho et al., 2021; Mao et al., 2023) employs multi-input and multi-output U-Net architectures (Ronneberger et al., 2015). Transformer-based deblurring techniques (Zamir et al., 2022; Wang et al., 2022; Tsai et al., 2022; Kong et al., 2023) are also developed, aiming to enhance image detail recovery by addressing the quadratic complexity of self-attention mechanisms in innovative ways. Additionally, several studies focus on network architecture design to effectively capture both local and global information (Zamir et al., 2021; Tu et al., 2022; Zamir et al., 2022; Li et al., 2023). Recently, prior-based methods (Li et al., 2022; Fang et al., 2023; Kim et al., 2024) have been investigated to further improve deblurring performance. NAFNet (Chen et al., 2022) introduces an efficient network architecture that utilizes simplified channel attention and simple gates. This scheme achieves state-of-the-art performance on the single image deblurring with small computational costs. Our method is based on this efficient kernel-free method but additionally employs a self-generated reference image, which enables us to further reduce computational costs while preserving state-of-the-art performance.

3 Reference-Based Blind Motion Deblurring

In this section, we present a new deblurring approach that leverages both kernel-based and kernel-free techniques. We begin by discussing a new approach to handling non-uniform deblurring tasks in Section 3.1. Next, we introduce a logarithmic fourier kernel estimator that yields a reference image in Section 3.2. Section 3.3 covers a reference-based kernel-free method, and finally, we present our hybrid method with implementation details in Section 3.4.

3.1 A new approach to handling non-uniform deblurring

The usage of additional reference images, which aid in restoring the texture details of the image, is a key component of our deblurring scheme. In essence, the image deblurring approaches (Zamir et al., 2021; Cho et al., 2021; Chen et al., 2022) aim to primarily recover the texture details of the image only from a blur image. On the other hand, our hybrid scheme focuses on recovering the texture details of the image from both a blur and a self-generated reference image. To this end, we first investigate a kernel-based method that follows a blurring process to generate a rich-detailed reference image inevitably with some artifacts. Then, our kernel-free model concentrates on getting rid of the artifacts in such reference images while preserving the remaining texture details.

In the case of uniform deblurring, given a pair of blur image y and uniform blur kernel k, the sharp image x is reconstructed by using the following deconvolution process

$$x = y \odot k, \tag{1}$$

where $\odot$ is a deconvolution operation. To extend it to non-uniform deblurring, let the total per-pixel error $e = e_k + e_m$ represent the kernel type error e_k (residual blurs) when the motion blur is not uniform, i.e., non-uniform blur, and the kernel model error e_m (artifacts) when the estimated kernel is not accurate. Then, we approximate a non-uniform deblurred image $\tilde{x}$,

$$\begin{aligned} \tilde{x} &= (y \odot k) + e_k = (y \odot (\tilde{k} + \Delta k)) + e_k \\ &= y \odot \tilde{k} + y \odot \Delta k + e_k = y \odot \tilde{k} + e_m + e_k \\ &= y \odot \tilde{k} + e = \tilde{x}_r + e, \end{aligned} \tag{2}$$

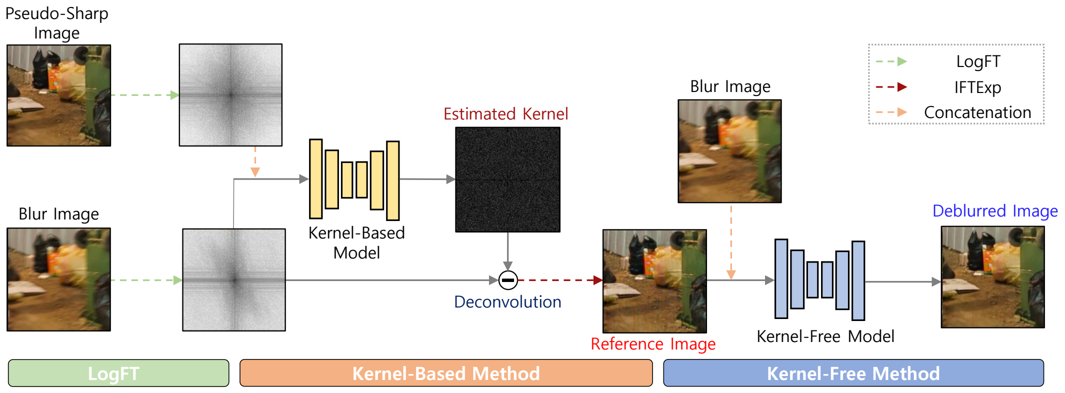


Figure 3: Network architecture overview. Our network architecture consists of two components: kernel-based model and kernel-free model. We use U-Net (Ronneberger et al., 2015) as our kernel-based model. NAFNet (Chen et al., 2022) is adopted for our kernel-free model. LogFT indicates the logarithmic fourier transform and IFTExp means the inverse logarithmic fourier transform.

where $\tilde{x}_r$ is the reference image and $\Delta k = k - \tilde{k}$ is the difference between real and estimated kernels. Since the kernel model error e_m mathematically contains the blur image in equation 2, we consider a kernel-free model f_θ that takes two inputs, e.g., reference image $\tilde{x}_r$ and blur image y, to handle two types of errors, e.g. $e = f_\theta(\tilde{x}_r, y)$. According to equation 2, the non-uniform deblurring problem (i.e., difficult task) is decomposed into uniform kernel estimation (reference estimation) and error prediction, i.e., two simpler tasks. Namely, the non-uniform deblurring can be viewed as reference-based deblurring with a self-generated reference image. To simplify the deconvolution process, we first consider a Discrete Fourier Transform (DFT) of a sample image y as follows:

$$Y(\omega) = \sum_{n=0}^{N-1} y(n) e^{-j\frac{2\pi}{N}\omega n}, \tag{3}$$

where $y(n)$ is a real-valued number at a pixel n and $Y(\omega)$ denotes a complex-valued number at a frequency ω. We remark that discrete fourier transform F naturally converts deconvolution into division, i.e., $y \odot k \leftrightarrows F(y)/F(k)$ (Brigham & Morrow, 1967). To further simplify the relationship between two fourier samples, we introduce a logarithmic operation in the fourier domain, leading to the logarithmic fourier transform F_L of the sample images, and then we obtain the relation $y \odot k \leftrightarrows F_L(y) - F_L(k)$ (Childers et al., 1977).

Let $X^L = F_L(x)$ and $Y^L = F_L(y)$ from a dataset $\mathcal{D} = \{(x, y)\}$ be sharp and blur samples by the logarithmic fourier operator F_L. Let $K^L = F_L(k)$ be a blur kernel in the logarithmic fourier space. We are interested in estimating kernel $\tilde{K^L}$ modeled by a kernel-based model g_ψ that maps from (X^L, Y^L) to $\tilde{K}$: we take two inputs, e.g., blur Y^L and sharp image X^L to facilitate estimating accurate kernels as if some literature considers image prior terms (Krishnan et al., 2011) for similar reasons. As the ground-truth sharp image is only valid in the training phase, we instead use the pseudo-sharp image by an off-the-shelf image deblurring model in the evaluation phase. Then, with the estimated kernel $\tilde{K}^L$, the reference image $\tilde{x}_r$ is generated by using simple subtraction (e.g., deconvolution) and inverse logarithmic fourier transform F_L^{-1}, i.e., $\tilde{x}_r = F_L^{-1}(Y^L - \tilde{K^L})$, as shown in Fig. 2 (c). Finally, we obtain the deblurred image $\tilde{x} = \tilde{x}_r + e$ where $\tilde{x}_r$ is the reference image estimated from our kernel-based model g_ψ and e is the total per-pixel error obtained by our kernel-free model f_θ. The whole network architecture is illustrated in Fig. 3.

3.2 Logarithmic fourier kernel estimation

In this section, two key schemes are presented to reduce the nature of ill-posedness (make it more accurate) and train a kernel estimator g_ψ under a light-weight network architecture (make it more efficient): (1) we utilize the logarithmic fourier space to learn a blur kernel more easily (Kim et al., 2024) , and (2) we exploit an additional pseudo-sharp image to predict a more accurate kernel. The logarithmic fourier space facilitates learning the blur kernel more easily by simplifying the relationship between blur and sharp samples.

Therefore, a light-weight network architecture may be sufficient to accommodate the simplified relation. Furthermore, since non-uniform kernel estimation may require a large network capacity as in (Gong et al., 2017; Zhang et al., 2021; Carbajal et al., 2023), we approximate a non-uniform kernel estimation to a uniform kernel estimation combined with the per-pixel error prediction as discussed in Section 3.1. Specifically, we estimate a uniform kernel under a small-scale network architecture. Then, the per-pixel errors caused by the artifacts and residual blurs are handled in our kernel-free method as explained in Section 3.3. Finally, using an additional pseudo-sharp image restricts the number of feasible kernel solutions, thereby making it simpler to estimate more accurate kernels (i.e., resulting in better references) as discussed in Section B.1.

To sum up, we aim to train a uniform kernel estimation model g_ψ using the logarithmic fourier pair of input samples, e.g., ground-truth sharp sample [1] X^L and blur sample Y^L. Our kernel-based model g_ψ outputs a logarithmic fourier uniform kernel that is expressed as

$$\tilde{K}^L = g_\psi(X^L, Y^L). \tag{4}$$

With the estimated kernel $\tilde{K}^L$, we can easily obtain a self-generated reference image as shown in Fig. 2 (c) by

$$\tilde{x}_r = F_L^{-1}(Y^L - g_\psi(X^L, Y^L)). \tag{5}$$

On the other hand, by taking the simple addition in the logarithmic fourier space, we have

$$\tilde{y} = F_L^{-1}(X^L + g_\psi(X^L, Y^L)), \tag{6}$$

where $\tilde{y}$ is an estimated blur image as shown in Fig. 2 (b). Finally, we suggest minimizing the following loss that coincides with the blurring and deblurring procedures to estimate the accurate kernels:

$$L_{\texttt{kernel}}(\psi; \mathcal{D}) = \frac{1}{|\mathcal{D}|} \sum_{(x,y)\in\mathcal{D}} d(\tilde{y}, y) + \lambda d(\tilde{x}_r, x), \tag{7}$$

where d is some distance or divergence in the image domain, e.g., the PSNR loss (Chen et al., 2022) and λ is a hyperparameter to control the sharpness of the reference image.

A higher value of λ results in an artifact-free reference image with a lack of texture details since it tends to aim for a high PSNR performance of the reference image. Meanwhile, a smaller value of λ focuses more on learning a motion kernel to accurately estimate a blur image $\tilde{y}$ by equation 6. Hence, with this motion kernel, the reference image becomes sharper but with some ringing artifacts due to the low contribution of the reference loss term $d(\tilde{x}, x)$ in equation 7. Moreover, the reference image trained with $\lambda = 0.01$ as shown in Fig. 1 (d) reveals higher texture details in comparison with the reference image trained with $\lambda = 0.1$ as shown in Fig. 1 (c). Further analysis will be explored in Section 5.

3.3 Kernel-free method with a reference image

To obtain better performance, it is common to scale up a network (He et al., 2016; Zagoruyko & Komodakis, 2016; Tan & Le, 2019) or use texture-detailed reference images (Zhang et al., 2019; Jiang et al., 2021). This section introduces our kernel-free method that exploits such texture-detailed reference images. Our kernel-free method eliminates the per-pixel artifacts (kernel model error e_m) and residual blurs (kernel type error e_k) using blur and self-generated reference images, as discussed in Section 3.1. Note that our kernel-free method can eliminate the per-pixel residual blurs that are still present in the reference images, especially when dealing with non-uniform blur types.

Given two input images, i.e., reference image $\tilde{x}_r$ and blur image y, the deblurred image $\tilde{x_\theta}$ is estimated via our kernel-free model f_θ as follows:

$$\tilde{x_\theta} = \tilde{x}_r + f_\theta(\tilde{x}_r, y), \tag{8}$$

[1] We observe that learning our kernel-based model with ground-truth sharp images results in better performance than with pseudo-sharp images.

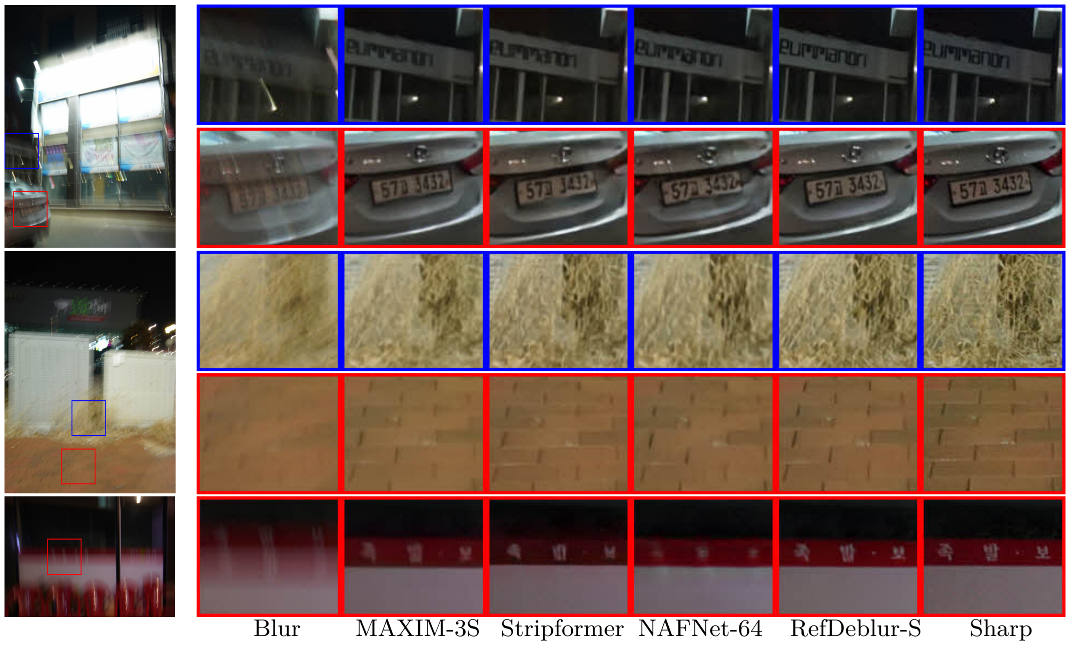


Figure 4: Visual comparison results on RealBlur-J (Rim et al., 2020) dataset. We compare our method with state-of-the-art deblurring methods (Cho et al., 2021; Mao et al., 2023; Tu et al., 2022; Chen et al., 2022; Tsai et al., 2022). The proposed method shows more texture-detailed, artifact-free, and sharper aspects in the deblurred images.

where $f_\theta(\tilde{x}_r, y)$ is the estimated per-pixel error $\tilde{e}$, $\tilde{x_\theta}$ denotes the final deblurred image as shown in Fig. 2 (d) and $\tilde{x}_r$ is the reference image defined by equation 5. Then, we derive our deblur loss that minimizes the distance between the real sharp and deblurred images by

$$L_{\texttt{deblur}}(\theta; \mathcal{D}) = \frac{1}{|\mathcal{D}|} \sum_{(x,y)\in\mathcal{D}} d(\tilde{x_\theta}, x), \tag{9}$$

where d is some distance or divergence in the image domain, e.g., the PSNR loss (Chen et al., 2022).

We observe that the reference image with high-frequency details generated from our kernel-based model as shown in Fig. 1 (d) achieves a PSNR performance of the final deblurred image (26.66 dB), which is better than the reference image with a lack of high-frequency details generated from the recent kernel-free model (Chen et al., 2022) as shown in Fig. 1 (b), which achieves a performance of 25.64 dB. This observation is consistent with the results as reported in Table 7, in which the reference image extracted from the recent kernel-free model(e.g., NAFNet (Chen et al., 2022)) results only in marginally improved performance. Furthermore, ℓ_1, ℓ_2, and PSNR losses are basically specialized to remove noises or artifacts rather than restore texture details. The proposed method is more relevant to use those losses since our kernel-free method is designed for removing artifacts of the texture-detailed reference image.

3.4 Reference-based blind motion deblurring

In this section, we present the implementation details of our hybrid method. Basically, our hybrid model consists of two models: kernel-based and kernel-free models. We adopt NAFNet (Chen et al., 2022) as our kernel-free model. We use U-Net (Ronneberger et al., 2015) as our kernel-based model because it empirically shows the consistent performance in various datasets. Since both models take two input images, we simply concatenate them toward the channel dimension to use the original network architectures. In the first stage, we train our kernel-based model g_ψ with the blur and ground-truth sharp images by minimizing equation 7. In the second stage, we freeze our kernel-based model g_ψ and optimize our kernel-free model f_θ with the reference $\tilde{x}_r$ (from our kernel-based model) and blur images y by using equation 9. To generate reference

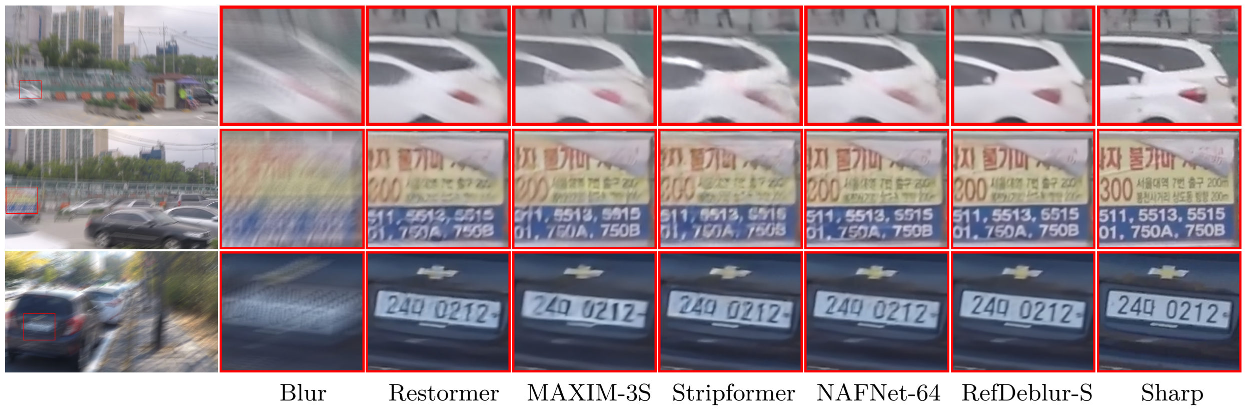


Figure 5: Visual comparison results on GoPro (Nah et al., 2017) dataset. We compare our method with state-of-the-art methods (Zamir et al., 2022; Tu et al., 2022; Tsai et al., 2022; Chen et al., 2022).

images, the ground-truth sharp images are used in the first stage whereas the pseudo-sharp image is used in the second stage: a pseudo-sharp image is obtained by an off-the-shelf image deblurring model, e.g., NAFNet (Chen et al., 2022).

To better generate the reference images for non-uniform blur types (i.e., further reducing the residual blurs in the reference images), we apply the local uniform approximation (LUA) to our kernel-based method at test time. Unlike (Whyte et al., 2014), we extract *non-overlapped* patches and then estimate patch-based blur kernels, whose complexity is identical to the kernel estimation of the full image. In this case, it may cause boundary artifacts when constructing the full reference image as shown in Fig. 6 (b). Since our kernel-free model is designed to remove various artifacts, the boundary artifacts can be removed in the final deblurred image as shown in Fig. 6 (c). Accordingly, our kernel-based method becomes more powerful in generating meaningful reference images from non-uniform blur images as exemplified in Fig. 6, such that it leads to better final deblurring performance as discussed in Section 5.

4 Experiments

4.1 Experimental setup

Dataset and evaluation metrics. We use GoPro (Nah et al., 2017), RealBlur (Rim et al., 2020), and RSBlur (Rim et al., 2022) for training and test sets. GoPro contains 2,103 and 1,111 blur-sharp image pairs for training and test sets, respectively. GoPro has been widely used for motion deblurring tasks, but it is a synthetic dataset where the blur image is generated by averaging sharp video frames captured by a high-speed camera. On the other hand, RealBlur and RSBlur are realistic motion blur datasets where they capture blur and sharp images in the same scene by using the beam splitter. RealBlur consists of RealBlur-J (sRGB domain) and RealBlur-R (RAW domain). Each RealBlur type comprises 3,758 and 980 image pairs for training and test sets, respectively. RSBlur dataset contains 8,878 and 3,360 blur-sharp image pairs for training and test sets, respectively. To verify the performance of the proposed method, we measure distortion metrics (e.g., Peak Signal to Noise Ration (PSNR) and Structural SIMilarity (SSIM) (Wang et al., 2004)). To measure model efficiency, we compute the number of network parameters and Multiply–ACcumulate operations (MACs) based on the image size of 256×256.

Network architecture variants. The default number of blocks of NAFNet (Chen et al., 2022) is 36. NAFNet with 16 widths, 32 widths, and 64 widths are referred to as NAFNet-16, NAFNet-32, and NAFNet-64, respectively. Similarly, we refer to our method with 16 widths as RefDeblur-16. To be consistent with the computational cost of NAFNet, we make RefDeblur-T (Tiny) and RefDeblur-S (Small). To cope with large-scale models such as Restormer (Zamir et al., 2022), MAXIM (Tu et al., 2022), and Stripformer (Tsai et al., 2022), we further increase the model sizes, which is referred to as RefDeblur-B (Big). Our network variants are described in details in Section A of Appendix.

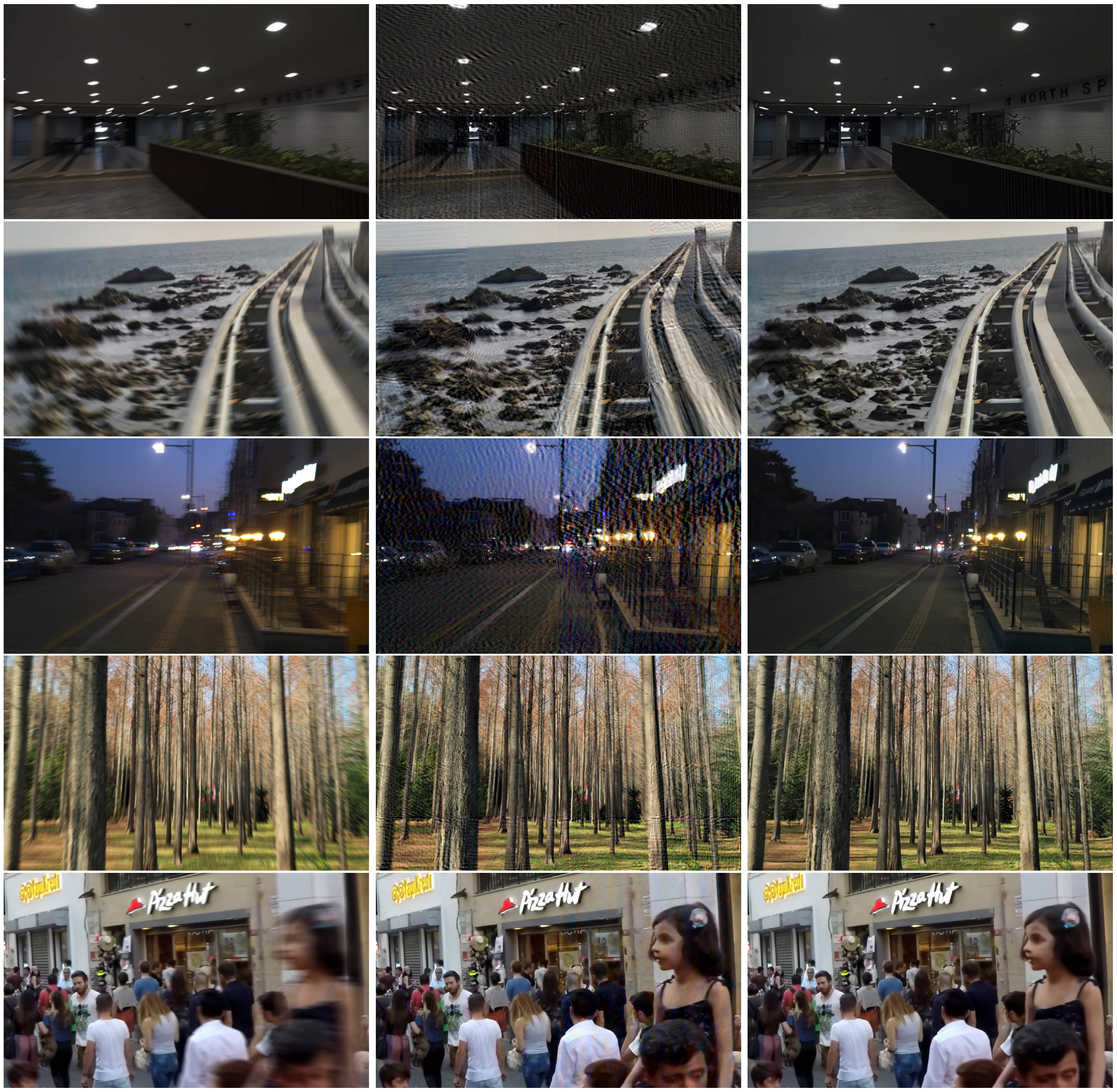


Figure 6: Visual results of blur, reference and final deblurred images. Starting from the top row, in order, saturation blur / in-plane rotation blur (e.g., z-axis rotation) / out-of-plane rotation blur (e.g., y-axis rotation) / forward motion blur (e.g., z-axis translation) / object motion blur.

Implementation details. We train with RealBlur (Rim et al., 2020), GoPro (Nah et al., 2017), and RSBlur (Rim et al., 2022) randomly cropped by 256×256. We train RefDeblur-16 up to $1,000$ epochs (batch size 16) for RealBlur, our RefDeblur variants (T / S / B) up to $2,000$ epochs (batch size 32) for RealBlur, our RefDeblur variants (S / B) up to $1,000$ epochs (batch size 32) for RSBlur and our RefDeblur-B up to $12,000$ epochs (batch size 64) for GoPro. Our kernel-free model is optimized by the AdamW (Loshchilov & Hutter, 2019) algorithm ($\beta_1 = 0.9$, $\beta_2 = 0.9$ and weight decay $1e^{-3}$) with the cosine annealing schedule ($1e^{-3}$ to $1e^{-7}$) gradually reduced for total iterations of each dataset. We optimize our kernel-based model using the Adam algorithm ($\beta_1 = 0.9$, $\beta_2 = 0.9$ and weight decay 0) with the step schedule (initial learning rate $1e^{-4}$) by decreasing the factor of 0.5 every 200 epochs for RealBlur and RSBlur dataset (up to $1,000$ epochs), and every 600 epochs for GoPro dataset (up to $3,000$ epochs). We employ the hyperparameter of

Table 1: The effect of the proposed method compared against NAFNet (Chen et al., 2022). The proposed method yields comparable performance with almost a quarter of the computational costs. Under similar computational costs, our method outperforms NAFNet with respect to several scales of network architectures.

Methods	# Params	GMACs	RealBlur-J	
			PSNR ↑	SSIM ↑
NAFNet-16 (Chen et al., 2022)	4.00	4.23	31.64	0.912
RefDeblur-16 (ours)	8.39	9.65	32.37	0.921
NAFNet-32 (Chen et al., 2022)	16.00	16.25	32.17	0.920
RefDeblur-T (ours)	12.40	16.33	32.70	0.927
NAFNet-64 (Chen et al., 2022)	63.50	63.64	32.66	0.929
RefDeblur-S (ours)	54.20	62.61	33.38	0.938
RefDeblur-B (ours)	123.50	136.88	**33.81**	**0.941**

Table 2: Comparison results on RealBlur-J and R datasets (Rim et al., 2020).

Methods	GMACs	RealBlur-J		RealBlur-R	
		PSNR↑	SSIM↑	PSNR↑	SSIM↑
MIMO-UNet+ (Cho et al., 2021)	154.41	31.92	0.919	-	-
MSDI-Net (Li et al., 2022)	336.43	32.35	0.923	-	-
MPRNet (Zamir et al., 2021)	777.01	31.76	0.922	39.31	0.972
Stripformer (Tsai et al., 2022)	169.89	32.48	0.929	39.84	0.974
MAXIM-3S (Tu et al., 2022)	169.50	32.84	0.935	39.45	0.962
FFTFormer (Kong et al., 2023)	131.45	32.62	0.932	40.20	0.973
GRL-B (Li et al., 2023)	1285.28	32.82	0.932	40.20	0.974
UFPNet (Fang et al., 2023)	243.33	33.35	0.934	40.61	0.974
FMIMO-UNet (Mao et al., 2023)	80.21	32.65	0.931	40.01	0.972
SefDeblur-L (Kim et al., 2024)	62.68	32.95	0.934	40.21	0.975
NAFNet-64 (Chen et al., 2022)	63.64	32.66	0.928	39.59	0.973
RefDeblur-S (ours)	62.61	33.38	0.938	40.70	0.976
RefDeblur-B (ours)	136.88	**33.81**	**0.941**	**41.16**	**0.978**

ours as $\lambda = 0.01$ unless otherwise specified. We adopt LUA as default, as discussed in Section 3.4. We use RealBlur-J for all ablation studies.

4.2 Reference images for various blur types

We examine that our kernel-based method produces reasonable reference images on several non-uniform blur types such as saturation, in-plane camera rotation (e.g., z-axis rotation), out-of-plane camera rotation (e.g., y-axis rotation), forward motion (e.g., z-axis translation) and object motion blurs. We adopt the local uniform approximation (LUA) in our kernel-based method to handle the non-uniform blur types effectively. To generate the reference images from the various non-uniform blur types, we collect those blur images from LOL-Blur (Zhou et al., 2022), Heterogeneous Motion Blur (Gong et al., 2017), GoPro (Nah et al., 2017), and real-world blur images captured by Samsung Galaxy Note 20 Ultra. As shown in Fig. 6 (b), under various non-uniform blur types, the reference images represent the sharpened visual results with some entailing artifacts. As discussed in Section 1, our reference images exhibit more ringing artifacts, especially in saturation and low-light (e.g., noisy) regions where the blur kernels become less accurate. Furthermore, the boundary artifacts by LUA are found in the reference images. Nevertheless, all those artifacts are removed by our kernel-free model as shown in Fig. 6 (c).

4.3 Effects of reference-based image deblurring

Model efficiency. This experiment aims to show the model efficiency of our method, compared to our baseline, e.g., NAFNet (Chen et al., 2022). We train and evaluate with RealBlur-J dataset (Rim et al., 2020). We measure PSNR and SSIM for model performance while the number of network parameters and MACs

Table 3: Comparison results on RSBlur (Rim et al., 2022) dataset. The best results are indicated in bold.

Methods	GMACs	RSBlur	
		PSNR ↑	SSIM ↑
SRN-Deblur (Tao et al., 2018)	1434.82	32.53	0.840
MIMO-UNet+ (Cho et al., 2021)	154.41	33.37	0.856
MPRNet (Zamir et al., 2021)	777.01	33.61	0.861
Restormer (Zamir et al., 2022)	141.00	33.69	0.863
Uformer-B (Wang et al., 2022)	89.50	33.98	0.866
IRNeXt (Cui et al., 2023)	114.79	34.08	0.869
ConvIR-L (Cui et al., 2024)	129.34	34.06	0.868
NAFNet-64 (Chen et al., 2022)	63.64	33.65	0.862
RefDeblur-S (ours)	62.61	34.01	0.867
RefDeblur-B (ours)	136.88	**34.24**	**0.871**

Table 4: Comparison results on GoPro (Nah et al., 2017) dataset. The best results are indicated in bold.

Methods	GMACs	GoPro	
		PSNR ↑	SSIM ↑
SRN-DeblurNet (Tao et al., 2018)	1434.82	30.26	0.932
DeblurGAN-v2 (Kupyn et al., 2019)	411.34	29.55	0.934
MPRNet (Zamir et al., 2021)	777.01	32.66	0.959
MIMO-UNet+ (Cho et al., 2021)	154.41	32.45	0.957
Restormer (Zamir et al., 2022)	141.00	32.92	0.961
Uformer-B (Wang et al., 2022)	89.50	32.97	0.967
Stripformer (Tsai et al., 2022)	169.89	33.08	0.962
MAXIM-3S (Tu et al., 2022)	169.50	32.86	0.961
FMIMO-UNet (Mao et al., 2023)	80.21	33.08	0.962
IRNeXt (Cui et al., 2023)	114.79	33.16	0.962
ConvIR-L (Cui et al., 2024)	129.34	33.28	0.963
NAFNet-64 (Chen et al., 2022)	63.64	33.04	0.967
NAFNet-96 (Chen et al., 2022)	141.72	33.02	0.967
RefDeblur-B (ours)	136.88	**33.60**	**0.970**

are measured for model efficiency. Note that the computational cost of FFT operation is negligibly minor so it is not considered in MACs. As shown in Table 1, the performance of our RefDeblur-T is comparable to that of NAFNet-64, almost with a quarter of the computational costs. Furthermore, our method improves PSNR from 32.66 to 33.38 dB with similar MACs (see the results of NAFNet-64 and RefDeblur-S).

Quantitative and qualitative results. We first present the quantitative results on RealBlur-J (Rim et al., 2020). The experimental results are summarized in Table 2. Our RefDeblur-B achieves the best performance, reaching 33.81 dB@PSNR, outperforming previous methods by a significant performance gap. Furthermore, as our method can be interpreted as prior-based deblurring method due to using an additional reference image, we compare it with other prior-based deblurring methods such as MSDI-Net (Li et al., 2022), UFPNet (Fang et al., 2023), and SegDeblur (Kim et al., 2024). We observe that our RefDeblur-S achieves superior deblurring performance compared to the prior-based methods, while requiring small computational cost (62.61 GMACs). As illustrated in Fig. 4, our method shows more texture-detailed and sharper images compared to state-of-the-art deblurring methods. Furthermore, we present qualitative results on real-world scenarios as shown in Fig. 7 of Appendix.

Results on RAW blur images. The realistic deblurring process defined by equation 1 may not hold in the sRGB domain since the acquired RAW blur images (e.g., RealBlur-R (Rim et al., 2020)) are non-linearly transformed to sRGB blur images (e.g., RealBlur-J (Rim et al., 2020)) via Image Signal Processing (ISP) pipelines (Rim et al., 2022). In other words, the RAW domain is a more relevant domain to predict more accurate blur kernels. This implies that our method can produce higher-quality reference images in the RAW

Table 5: Ablation study on our components. PSNR with LUA means we apply the local uniform approximation in our kernel-based method. Experimental results show that all components are necessary and positively combined to provide the best performance as indicated in bold.

(i) Logarithmic Fourier Transform				✓
(ii) Our Kernel-Based Model			✓	✓
(iii) Training with Reference		✓	✓	✓
PSNR (dB)	31.64	31.73	31.89	**32.14**
PSNR with LUA (dB)	-	-	32.20	**32.37**

domain, leading to further performance improvement. To verify this, we train and evaluate with RealBlur-R. As shown in Table 2, our method significantly improves PSNR performance from 39.59 (NAFNet-64) to 40.70 dB (RefDeblur-S) under similar computational costs. Interestingly, the performance gap in the RAW domain, i.e., RealBlur-R (1.11 dB) between NAFNet-64 and RefDeblur-S is noticeably higher than the gap in the sRGB domain, i.e., RealBlur-J (0.72 dB). Since the blur kernels are more accurately estimated in the RAW domain, the RAW domain is the best demonstration of the effectiveness of our method.

4.4 Comparison to kernel-free methods

Performance on realistic high-resolution blur dataset. To compare with the recent works under the high-resolution (around 1920×1200) realistic blur dataset, we train and evaluate with RSBlur (Rim et al., 2022). As shown in Table 3, our RefDeblur-B achieves the best performance over the previous methods. In particular, our RefDeblur-S improves PSNR performance of NAFNet-64 from 33.65 to 34.01 dB under similar computational costs.

Reference-driven model scalability. To investigate the model scalability with our reference image, we compare our method with the recent kernel-free methods under GoPro (Nah et al., 2017). We scale up the model size from NAFNet-64 (63.64 GMACs) to NAFNet-96 (141.72 GMACs) and compare it with RefDeblur-B (136.88 GMACs). As shown in Table 4, the result shows that simply increasing the model size from NAFNet-64 to NAFNet-96 does not necessarily improve performance: it is also reported in (Chen et al., 2022) (see the result in Table 3 of the literature). However, our RefDeblur-B achieves 33.60 dB although its GMACs is similar to NAFNet-96. The performance improvement from NAFNet-96 (33.02 dB) to our RefDeblur-B (33.60 dB) is primarily attributed to the integration of our reference image. These results demonstrate the advantage of model scalability in our reference-based deblurring method compared to that of the kernel-free deblurring method. Furthermore, we observe that our method recovers the texture details of the images better than the recent kernel-free methods as illustrated in Fig. 5.

5 Ablation study

Functionality of each module. In this section, we explore which module of our method has more impact on performance improvement. As described in Table 5, with no module, it indicates the original NAFNet-16 (Chen et al., 2022). When adding (iii), it directly uses the pseudo-sharp image as the reference image. The method including (ii) and (iii) mean RefDeblur-16 but uses fourier space instead of logarithmic fourier space. The final method containing all modules denotes our RefDeblur-16. As shown in Table 5, we observe that adding the module such as (i) or (ii) provides a meaningful performance improvement from 31.64 to 31.89 dB and from 31.89 to 32.14 dB, respectively. Furthermore, simply introducing an additional reference image does not highly improve performance (see the results of the first and second columns). Finally, when applying LUA in our kernel-based method, we achieve a performance improvement from 32.14 to 32.37 dB (see PSNR and PSNR with LUA of the fourth column). This explains why our kernel-based method can be applicable to various non-uniform blur types.

Effects on λ. We investigate which hyperparameter λ produces the most suitable reference image. We experiment with $\lambda = \{0.001, 0.01, 0.1, 1.0\}$. Given the reference images varying λ from 0.001 to 1.0, the final deblurring results are shown in Table 6. Although the low λ such as 0.01 or 0.1 leads to some artifacts as shown in Fig. 6 (b), it provides a better rich-detailed reference image, thereby resulting in better deblurring

Table 6: Ablation study on hyperparameter λ. We investigate the hyperparameter λ since it is related to the quality of the reference image, which influences the final performance.

Methods	λ	PSNR ↑	SSIM ↑
NAFNet-16	-	31.58	0.912
RefDeblur-16	0.001	32.14	0.919
	0.01	**32.37**	**0.921**
	0.1	32.24	0.918
	1.0	32.20	0.916

Table 7: Ablation study on types of reference images.

Reference Model			Ref.	Deblur
Type	Loss	GMACs	PSNR	PSNR
NAFNet-16	PSNR	4.23	**31.64**	31.73
	VGG (Johnson et al., 2016)	4.23	29.56	31.76
	VGG+GAN (Goodfellow et al., 2014)	4.23	30.81	31.80
RefDeblur-16	PSNR	5.39	27.89	**32.37**
No Reference		-	-	31.58
J-MKPD (Carbajal et al., 2023)		56.36	28.88	31.32
MotionETR (Zhang et al., 2021)		55.92	28.04	31.28

performance. On the other hand, when λ is too small such as 0.001, it gives rise to a large amount of artifacts, so that such high dominance of artifacts in the reference image results in degraded performance despite its high-frequency details contained. Therefore, we choose the hyperparameter λ as 0.01 in our experiments.

Types of reference images. The goal of this experiment is to verify which types of reference images are appropriate. To generate various types of reference images, the pseudo-sharp images trained with PSNR, perceptual (VGG) (Johnson et al., 2016), and VGG + Generative Adversarial Networks (GAN) (Goodfellow et al., 2014) losses are chosen as reference images. Also, other kernel-based methods such as J-MKPD (Carbajal et al., 2023) and MotionETR (Zhang et al., 2021) are considered as reference generators. Note that we consider the VGG and VGG+GAN losses because they are widely studied to perceptually recover texture-detailed images in image restoration tasks (Johnson et al., 2016; Ledig et al., 2017). Given those reference images, we train our kernel-free models with them. The PSNR performance of reference and deblurred images are presented in Table 7. Although the reference images by VGG and VGG + GAN losses bring in better final deblurring results compared with the result of "No Reference", their performance improvements are relatively small when compared to our method: some texture details from VGG and VGG + GAN may appear in unpleasant visual artifacts rather than realistic details in the image domain (Liang et al., 2022). Furthermore, since the other methods aim for a high PSNR performance of the reference image, it does not help produce texture-detailed reference images. As a reference image, prioritizing the inclusion of high-frequency realistic details even with some artifacts is more crucial for achieving better final deblurring results compared to focusing solely on a higher reference PSNR (compare the results of "RefDeblur-16" with those of "NAFNet-16 with PSNR"). Also, it is proven that the reference image is not always helpful for final deblurring performance (compare the results between "No Reference" and "J-MKPD" and "MotionETR").

6 Conclusions

In this paper, we propose a reference-based motion deblurring scheme that combines kernel-based and kernel-free methods. Our method decomposes non-uniform deblurring into two tasks: a light-weight kernel-based model that estimates a blur kernel in logarithmic fourier space to produce a texture-rich reference image, and a kernel-free model that refines this image by correcting residual errors. We demonstrate that such dual strategy not only leverages the strengths of both methods but also reduces computational costs by 75%, compared to state-of-the-art methods. The proposed method can be extended to other deblurring tasks such as video deblurring (Wang et al., 2019) and defocus deblurring (Abuolaim & Brown, 2020).

References

Abdullah Abuolaim and Michael S Brown. Defocus deblurring using dual-pixel data. *The European Conference on Computer Vision (ECCV)*, pp. 111–126, 2020.

E Oran Brigham and RE Morrow. The fast fourier transform. *IEEE spectrum*, 4(12):63–70, 1967.

Guillermo Carbajal, Patricia Vitoria, José Lezama, and Pablo Musé. Blind motion deblurring with pixel-wise kernel estimation via kernel prediction networks. *IEEE Transactions on Computational Imaging*, 2023.

Ayan Chakrabarti. A neural approach to blind motion deblurring. *The European Conference on Computer Vision (ECCV)*, pp. 221–235, 2016.

Liangyu Chen, Xiaojie Chu, Xiangyu Zhang, and Jian Sun. Simple baselines for image restoration. *The European Conference on Computer Vision (ECCV)*, 2022.

Donald G Childers, David P Skinner, and Robert C Kemerait. The cepstrum: A guide to processing. *Proceedings of the IEEE*, 65(10):1428–1443, 1977.

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. *The IEEE International Conference on Computer Vision (ICCV)*, pp. 4641–4650, 2021.

Yuning Cui, Wenqi Ren, Sining Yang, Xiaochun Cao, and Alois Knoll. IRNeXt: Rethinking convolutional network design for image restoration. *International Conference on Machine Learning (ICML)*, pp. 6545–6564, 2023.

Yuning Cui, Wenqi Ren, Xiaochun Cao, and Alois Knoll. Revitalizing convolutional network for image restoration. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2024. doi: 10.1109/TPAMI.2024.3419007.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *International Conference on Learning Representations (ICLR)*, 2021.

Zhenxuan Fang, Fangfang Wu, Weisheng Dong, Xin Li, Jinjian Wu, and Guangming Shi. Self-supervised non-uniform kernel estimation with flow-based motion prior for blind image deblurring. *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 18105–18114, 2023.

D. A. Fish, A. M. Brinicombe, E. R. Pike, and J. G. Walker. Blind deconvolution by means of the richardson–lucy algorithm. *J. Opt. Soc. Am. A*, 12(1):58–65, 1995.

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. *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2319–2328, 2017.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. *Advances in Neural Information Processing Systems (NIPS)*, pp. 2672–2680, 2014.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

Yuming Jiang, Kelvin CK Chan, Xintao Wang, Chen Change Loy, and Ziwei Liu. Robust reference-based super-resolution via c2-matching. *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2103–2112, 2021.

Justin Johnson, Alexandre Alahi, and Li Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. *The European Conference on Computer Vision (ECCV)*, pp. 694–711, 2016.

Insoo Kim, Jae Seok Choi, Geonseok Seo, Kinam Kwon, Jinwoo Shin, and Hyong-Euk Lee. Real-world efficient blind motion deblurring via blur pixel discretization. *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 25879–25888, 2024.

Lingshun Kong, Jiangxin Dong, Mingqiang Li, Jianjun Ge, and Jinshan Pan. Efficient frequency domain-based transformers for high-quality image deblurring, 2023.

Dilip Krishnan and Rob Fergus. Fast image deconvolution using hyper-laplacian priors. *Advances in neural information processing systems (NIPS)*, 22, 2009.

Dilip Krishnan, Terence Tay, and Rob Fergus. Blind deconvolution using a normalized sparsity measure. *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 233–240, 2011.

Orest Kupyn, Volodymyr Budzan, Mykola Mykhailych, Dmytro Mishkin, and Jiří Matas. Deblurgan: Blind motion deblurring using conditional adversarial networks. *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 8183–8192, 2018.

Orest Kupyn, Tetiana Martyniuk, Junru Wu, and Zhangyang Wang. Deblurgan-v2: Deblurring (orders-of-magnitude) faster and better. *The IEEE International Conference on Computer Vision (ICCV)*, pp. 8878–8887, 2019.

Christian Ledig, Lucas Theis, Ferenc Huszár, Jose Caballero, Andrew Cunningham, Alejandro Acosta, Andrew Aitken, Alykhan Tejani, Johannes Totz, Zehan Wang, et al. Photo-realistic single image super-resolution using a generative adversarial network. *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 4681–4690, 2017.

Dasong Li, Yi Zhang, Ka Chun Cheung, Xiaogang Wang, Hongwei Qin, and Hongsheng Li. Learning degradation representations for image deblurring. *The European Conference on Computer Vision (ECCV)*, pp. 736–753, 2022.

Yawei Li, Yuchen Fan, Xiaoyu Xiang, Denis Demandolx, Rakesh Ranjan, Radu Timofte, and Luc Van Gool. Efficient and explicit modelling of image hierarchies for image restoration. *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023.

Jie Liang, Hui Zeng, and Lei Zhang. Details or artifacts: A locally discriminative learning approach to realistic image super-resolution. *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5657–5666, 2022.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *International Conference on Learning Representations (ICLR)*, 2019.

Xintian Mao, Yiming Liu, Fengze Liu, Qingli Li, Wei Shen, and Yan Wang. Intriguing findings of frequency selection for image deblurring. *Association for the Advancement of Artificial Intelligence (AAAI)*, 2023.

Seungjun Nah, Tae Hyun Kim, and Kyoung Mu Lee. Deep multi-scale convolutional neural network for dynamic scene deblurring. *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3883–3891, 2017.

Seungjun Nah, Sanghyun Son, Jaerin Lee, and Kyoung Mu Lee. Clean images are hard to reblur: Exploiting the ill-posed inverse task for dynamic scene deblurring. *International Conference on Learning Representations (ICLR)*, 2022.

Jaesung Rim, Haeyun Lee, Jucheol Won, and Sunghyun Cho. Real-world blur dataset for learning and benchmarking deblurring algorithms. *The European Conference on Computer Vision (ECCV)*, pp. 184–201, 2020.

Jaesung Rim, Geonung Kim, Jungeon Kim, Junyong Lee, Seungyong Lee, and Sunghyun Cho. Realistic blur synthesis for learning image deblurring. *The European Conference on Computer Vision (ECCV)*, 2022.

Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. *International Conference on Medical image computing and computer-assisted intervention*, pp. 234–241, 2015.

Christian J Schuler, Michael Hirsch, Stefan Harmeling, and Bernhard Schölkopf. Learning to deblur. *IEEE transactions on pattern analysis and machine intelligence*, 38(7):1439–1451, 2015.

Jian Sun, Wenfei Cao, Zongben Xu, and Jean Ponce. Learning a convolutional neural network for non-uniform motion blur removal. *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 769–777, 2015.

Mingxing Tan and Quoc Le. Efficientnet: Rethinking model scaling for convolutional neural networks. *International conference on machine learning (ICML)*, pp. 6105–6114, 2019.

Xin Tao, Hongyun Gao, Xiaoyong Shen, Jue Wang, and Jiaya Jia. Scale-recurrent network for deep image deblurring. *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 8174–8182, 2018.

Ilya O Tolstikhin, Neil Houlsby, Alexander Kolesnikov, Lucas Beyer, Xiaohua Zhai, Thomas Unterthiner, Jessica Yung, Andreas Steiner, Daniel Keysers, Jakob Uszkoreit, et al. Mlp-mixer: An all-mlp architecture for vision. *Advances in Neural Information Processing Systems (NeurIPS*, 34:24261–24272, 2021.

Phong Tran, Anh Tuan Tran, Quynh Phung, and Minh Hoai. Explore image deblurring via encoded blur kernel space. *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 11956–11965, 2021.

Fu-Jen Tsai, Yan-Tsung Peng, Yen-Yu Lin, Chung-Chi Tsai, and Chia-Wen Lin. Stripformer: Strip transformer for fast image deblurring. *The European Conference on Computer Vision (ECCV)*, 2022.

Zhengzhong Tu, Hossein Talebi, Han Zhang, Feng Yang, Peyman Milanfar, Alan Bovik, and Yinxiao Li. Maxim: Multi-axis mlp for image processing. *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5769–5780, 2022.

Xintao Wang, Kelvin CK Chan, Ke Yu, Chao Dong, and Chen Change Loy. Edvr: Video restoration with enhanced deformable convolutional networks. *The IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2019.

Zhendong Wang, Xiaodong Cun, Jianmin Bao, Wengang Zhou, Jianzhuang Liu, and Houqiang Li. Uformer: A general u-shaped transformer for image restoration. *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 17683–17693, 2022.

Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing*, 13(4):600–612, 2004.

Oliver Whyte, Josef Sivic, and Andrew Zisserman. Deblurring shaken and partially saturated images. *International journal of computer vision*, 110(2):185–201, 2014.

Lu Yuan, Jian Sun, Long Quan, and Heung-Yeung Shum. Image deblurring with blurred/noisy image pairs. *ACM SIGGRAPH 2007 papers*, pp. 1–es, 2007.

Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. *The British Machine Vision Conference (BMVC)*, 2016.

Syed Waqas Zamir, Aditya Arora, Salman Khan, Munawar Hayat, Fahad Shahbaz Khan, Ming-Hsuan Yang, and Ling Shao. Multi-stage progressive image restoration. *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 14821–14831, 2021.

Syed Waqas Zamir, Aditya Arora, Salman Khan, Munawar Hayat, Fahad Shahbaz Khan, and Ming-Hsuan Yang. Restormer: Efficient transformer for high-resolution image restoration. *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5728–5739, 2022.

Youjian Zhang, Chaoyue Wang, Stephen J Maybank, and Dacheng Tao. Exposure trajectory recovery from motion blur. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(11):7490–7504, 2021.

Zhifei Zhang, Zhaowen Wang, Zhe Lin, and Hairong Qi. Image super-resolution by neural texture transfer. *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 7982–7991, 2019.

Shangchen Zhou, Chongyi Li, and Chen Change Loy. Lednet: Joint low-light enhancement and deblurring in the dark. *The European Conference on Computer Vision (ECCV)*, pp. 573–589, 2022.

A Implementation Details

Logarithmic operation. For stable training, we add 10^{-12} when performing the complex logarithmic operation. Furthermore, we stabilize our kernel-based model by initializing the model weights to have the normal distribution (mean : 0, standard deviation: 0.02). Hence, the estimated kernel begins by the zero kernel in the logarithmic foruier domain (equivalent to the identity kernel in the spatial domain).

Network architecture variants. Our hybrid model consists of two models: kernel-based and kernel-free models. We adopt NAFNet (Chen et al., 2022) as our kernel-free model. We use U-Net (Ronneberger et al., 2015) as our kernel-based model because it empirically shows the consistent performance in various datasets. Also, we consider the pre-trained NAFNet for generating pseudo-sharp images used in our kernel-based model. The default number of blocks of NAFNet (Chen et al., 2022) is 36 which consists of encoder blocks $\{2, 2, 2, 20\}$, middle block $\{2\}$ and decoder blocks $\{2, 2, 2, 20\}$. NAFNet with 16 widths, 32 widths, and 64 widths are referred to as NAFNet-16, NAFNet-32, and NAFNet-64, respectively. Similarly, we refer to our method with 16 widths as RefDeblur-16 as shown in Table 8. To be consistent with the computational cost of NAFNet-32 and 64, we increase the model size of the pseudo-sharp model (NAFNet-16 $\rightarrow$ NAFNet-16 or NAFNet-32), our kernel-based (UNet-16 $\rightarrow$ UNet-32) and kernel-free model (NAFNet-16 $\rightarrow$ NAFNet-16+ or NAFNet-32+) in RefDeblur-16, which are denoted as RefDeblur-T and RefDeblur-S as shown in Table 8. To cope with the large-scale models such as MAXIM (Tu et al., 2022), Restormer (Zamir et al., 2022), and Stripformer (Tsai et al., 2022), we further increase the model size of the pseudo-sharp (NAFNet-32 $\rightarrow$ NAFNet-64) and kernel-free model (NAFNet-32+ $\rightarrow$ NAFNet-32++) in RefDeblur-S, which is referred to as RefDeblur-B as shown in Table 8. Here, NAFNet16+ consists of encoder blocks $\{6, 6, 6, 22\}$, middle block $\{6\}$ and decoder blocks $\{6, 6, 6, 6\}$ with 16 widths; NAFNet-32+ consists of encoder blocks $\{9, 9, 9, 25\}$, middle block $\{9\}$ and decoder blocks $\{9, 9, 9, 9\}$ with 32 widths; and NAFNet-32++ consists of encoder blocks $\{16, 16, 16, 32\}$, middle block $\{16\}$ and decoder blocks $\{16, 16, 16, 16\}$ with 32 widths. Note that we use a variant of U-Net that performs four times of downsampling by stride 2 operations. Overall, our models are summarized in Table 8.

Table 8: Our model variants. We present a detailed description of our individual variants such as the sub-models types and the computational costs. The individual GMACs of the sub-models are denoted in brackets.

Our Model	Model (GMACs)			Total GMACs	Total # Params
	Pseudo-Sharp	Kernel-Based	Kernel-Free		
RefDeblur-16	NAFNet-16 (4.23)	UNet-16 (1.16)	NAFNet-16 (4.26)	9.65	8.39
RefDeblur-T	NAFNet-16 (4.23)	UNet-32 (3.99)	NAFNet-16+ (8.11)	16.33	12.40
RefDeblur-S	NAFNet-32 (16.25)	UNet-32 (3.99)	NAFNet-32+ (42.37)	62.61	54.20
RefDeblur-B	NAFNet-64 (63.64)	UNet-32 (3.99)	NAFNet-32++ (69.25)	136.88	123.50

B Additional ablation study

B.1 Effects on pseudo-sharp images

In this section, we explore our kernel-based model to confirm whether the pseudo-sharp image is necessary. As discussed in Section 3.2, we use the pseudo-sharp image because the paired input can restrict the number of feasible kernel solutions, which results in estimating more accurate kernels and better final deblurring results. To confirm this, we train our kernel-based model with blur images only, with blur + ground-truth (GT) images, and with blur + pseudo-sharp images to produce the corresponding reference images. Then, we use those reference images to train our kernel-free models. As shown in Table 9, training our kernel-based model with blur image only leads to performance degradation compared with the no-reference result (e.g., NAFNet-16). This shows the difficulty to estimate the accurate kernels under a single blur input image that suffers from the huge ill-posedness. On the other hand, we can boost the final PSNR and SSIM performances when using pair input images such as blur + pseudo-sharp images and blur + ground-truth sharp images in our kernel-based model as depicted in Table 9. While generating the pseudo-sharp image consumes computational costs, our method maintains comparable overall computational efficiency to NAFNet, achieving better performance (see Table 1). Furthermore, we can intuitively expect that the reference image based on "Blur + GT" contains better realistic texture details than the others. Hence, the result of "Blur + GT" is the best demonstration that a better texture-detailed reference image leads to improved final PSNR results.

Table 9: Ablation study on input types of our kernel-based method. We investigate several input combinations such as a single blur input, pair input (blur + pseudo-sharp), and another pair input (blur + GT) in our kernel-based model. The best results are indicated in bold.

Architecture	Kernel-Based Input Types	Kernel-Free Model	
		PSNR	SSIM
NAFNet-16	-	31.64	0.912
RefDeblur-16	Blur Only	31.56	0.910
	Blur + GT	**34.81**	**0.935**
	Blur + Pseudo-Sharp	32.37	0.920

B.2 Effects on model size

Our hybrid model consists of kernel-based and kernel-free models. To obtain a pseudo-sharp image for our kernel-based model, an additional off-the-shelf pseudo-sharp model, e.g., pre-trained NAFNet (Chen et al., 2022) is considered. Here, we raise a fundamental question: which model size should we increase to improve performance? To address this, we conduct experiments with various model sizes by scaling the size of pseudo-sharp (PS), kernel-based (KB), and kernel-free (KF) models. The results are shown in Table 10. It is reported that the small-scale kernel-based model is sufficient to provide high performance (compare the performance in the 1st-2nd-3rd row results). Furthermore, we observe that the size of the kernel-free model is the most important ingredient to improve performance. This is witnessed by comparing the 2nd-3rd (KB size increase), 2nd-4th (PS size increase), and 2nd-5th (KF size increase) rows for total MACs and PSNR.

Table 10: Ablation study on model sizes of our method. We make several combinations of pseudo-sharp, kernel-based, and kernel-free models when varying the size of each model. The bolded GMACs indicate the increase compared to the previous row.

MACs (G)				PSNR	SSIM
Pseudo-Sharp	Kernel-Based	Kernel-Free	Total		
16.25	1.16	16.30	33.71	32.92	0.931
16.25	**3.99**	16.30	36.54	33.00	0.931
16.25	**14.69**	16.30	47.24	33.00	0.931
63.64	3.99	16.30	83.93	33.28	0.933
16.25	3.99	**42.37**	62.61	33.38	0.938
16.25	3.99	**69.25**	89.49	33.60	0.940

B.3 Spatial vs. Fourier vs. LogFourier

We examine whether the logarithmic fourier space is the most effective input domain in our kernel-based method. To this end, we train our kernel-based models using input samples in spatial, fourier, and logarithmic fourier domains to make reference images which are then used to train our kernel-free models. The final deblurring results are reported in Table 11. The results show that the logarithmic fourier space leads to the best deblurring performance among the domains since the logarithmic fourier domain enables to find more accurate blur kernels by simplifying the relationship between sharp and blur images (e.g., deconvolution to subtraction).

Table 11: Ablation study on input domains in our kernel-based method. The best results are indicated in bold.

Architecture	Kernel-Based Input Domain	Kernel-Free Model	
		PSNR	SSIM
NAFNet-16	-	31.58	0.912
RefDeblur-16	Spatial	31.74	0.913
	Fourier	32.20	0.919
	LogFourier	**32.37**	**0.920**

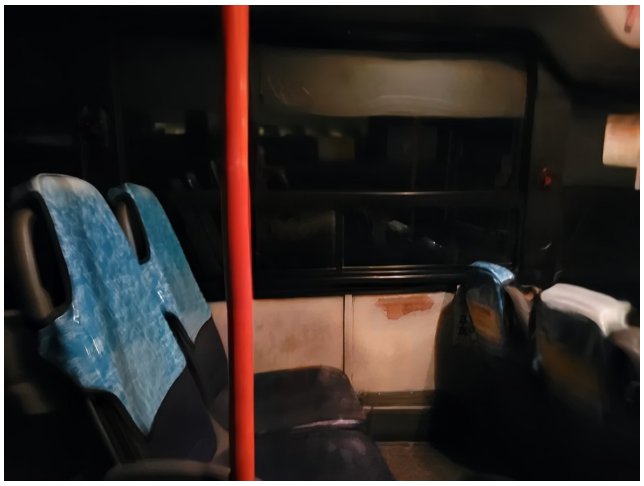
C Generalization to real-world blur images

We capture real-world blur images with natural hand motions by Samsung Galaxy Note 20 Ultra. We use our *RefDeblur-S* model trained with RealBlur-J (Rim et al., 2020) in order to show the effectiveness of our method under the lowest complexity, compared with the previous methods such as Stripformer (Tsai et al., 2022) and NAFNet-64 (Chen et al., 2022). As shown in Fig. 7, the real-world blur images are well-reconstructed by our method. Meanwhile, the compared methods work well on some blur images but some other blur images are not well-recovered. This shows a better generalization of our method compared with other methods.

(a) Blur Input

(b) NAFNet-64

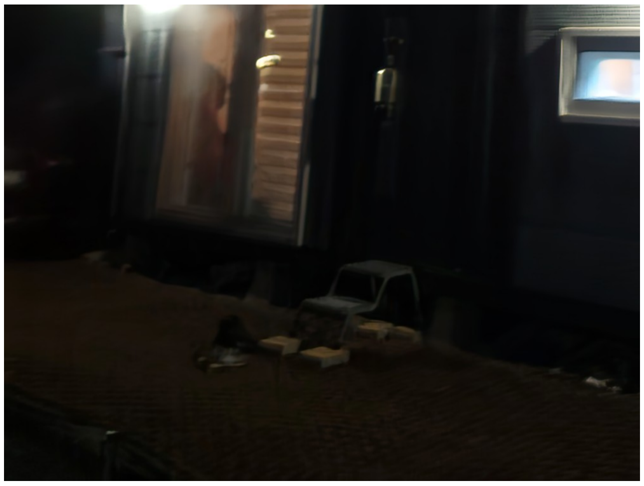
(c) Stripformer

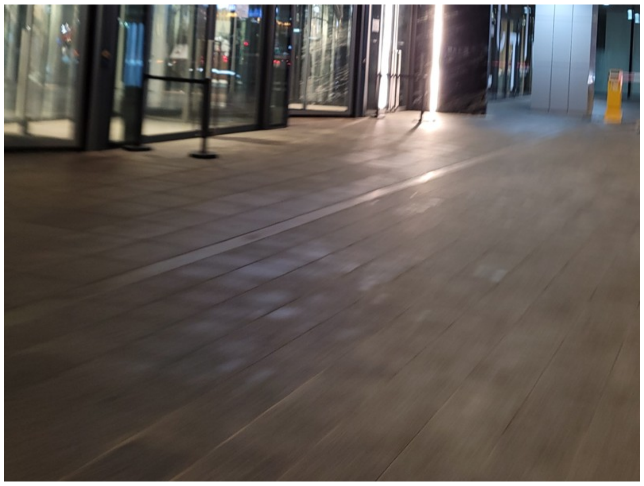

(d) RefDeblur-S (ours)

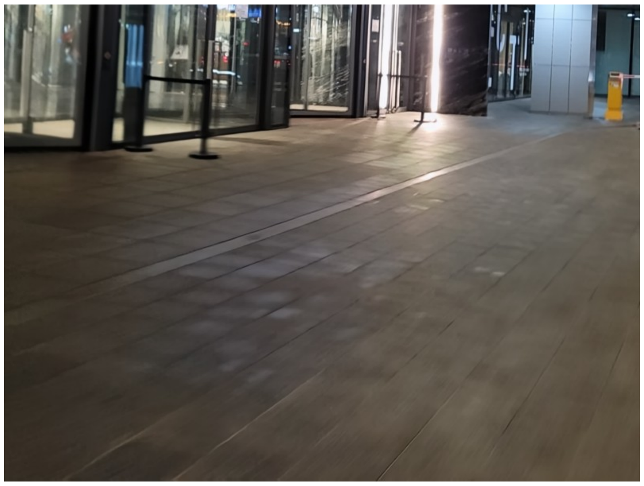
(a) Blur Input

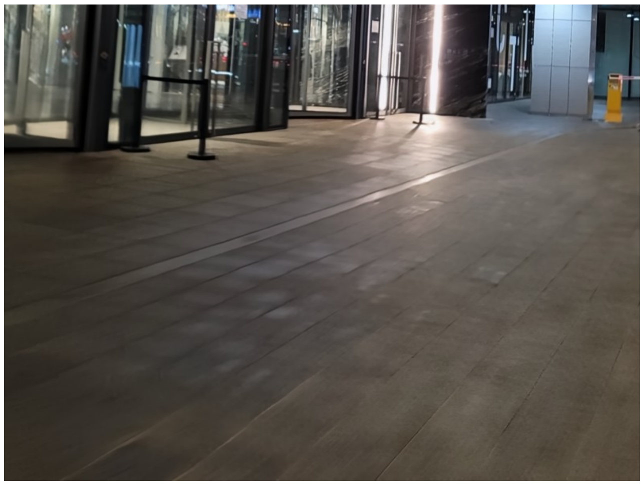
(b) NAFNet-64

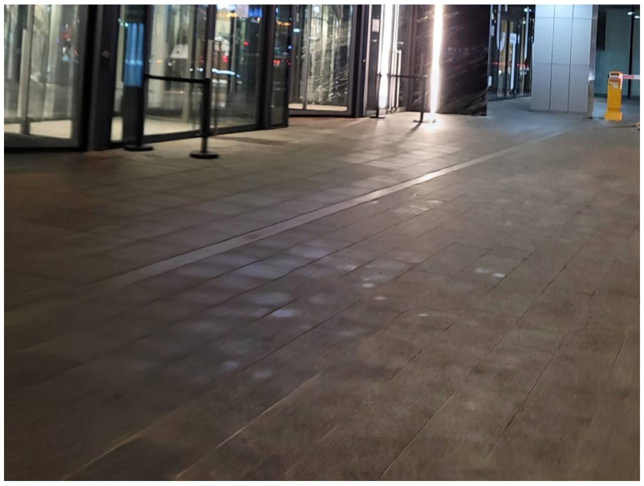
(c) Stripformer

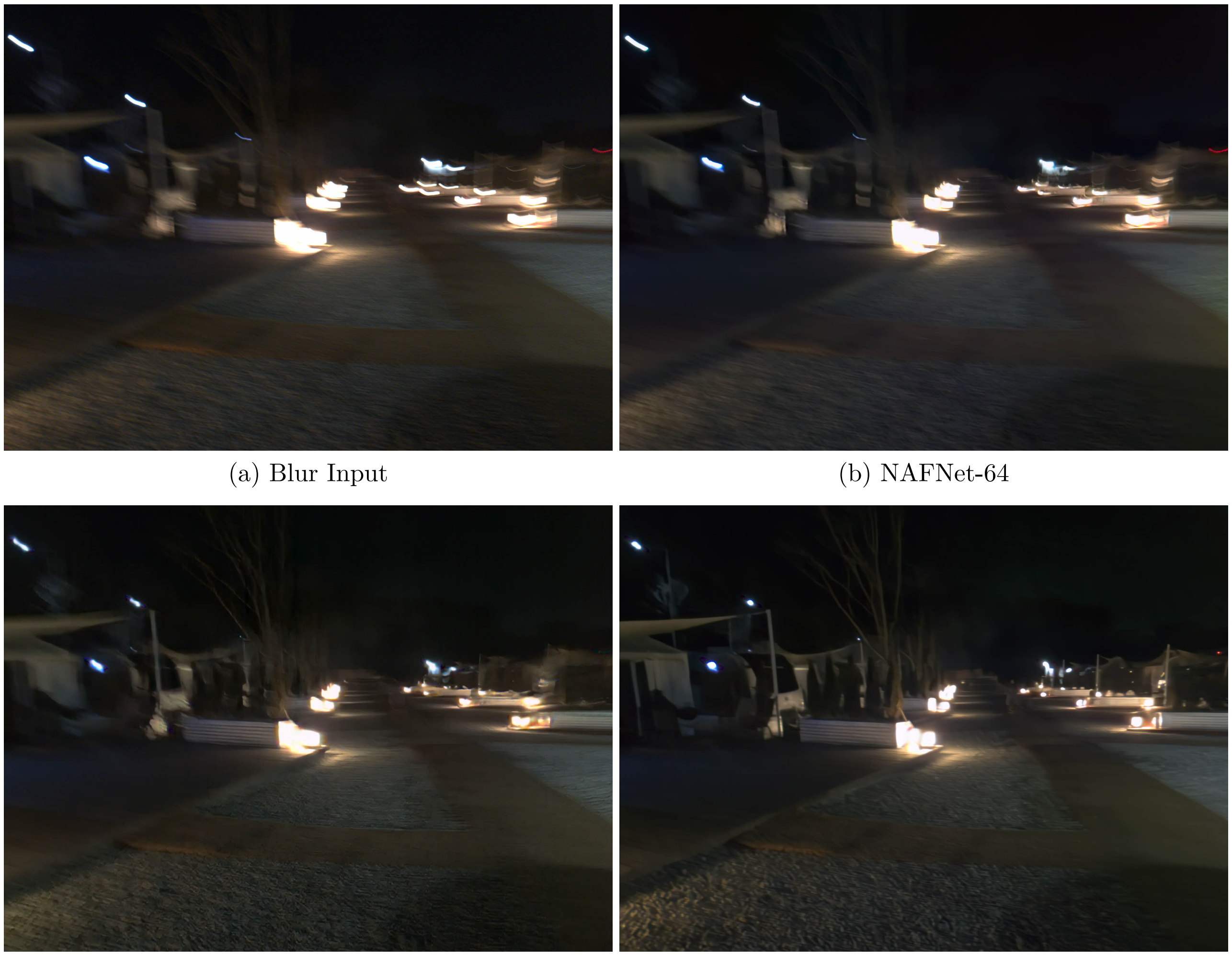

(d) RefDeblur-S (ours)


(a) Blur Input


(b) NAFNet-64


(c) Stripformer


(d) RefDeblur-S (ours)



(a) Blur Input

(b) NAFNet-64

(c) Stripformer

(d) RefDeblur-S (ours)

Figure 7: Qualitative results on real-world scenarios.

D Additional qualitative results

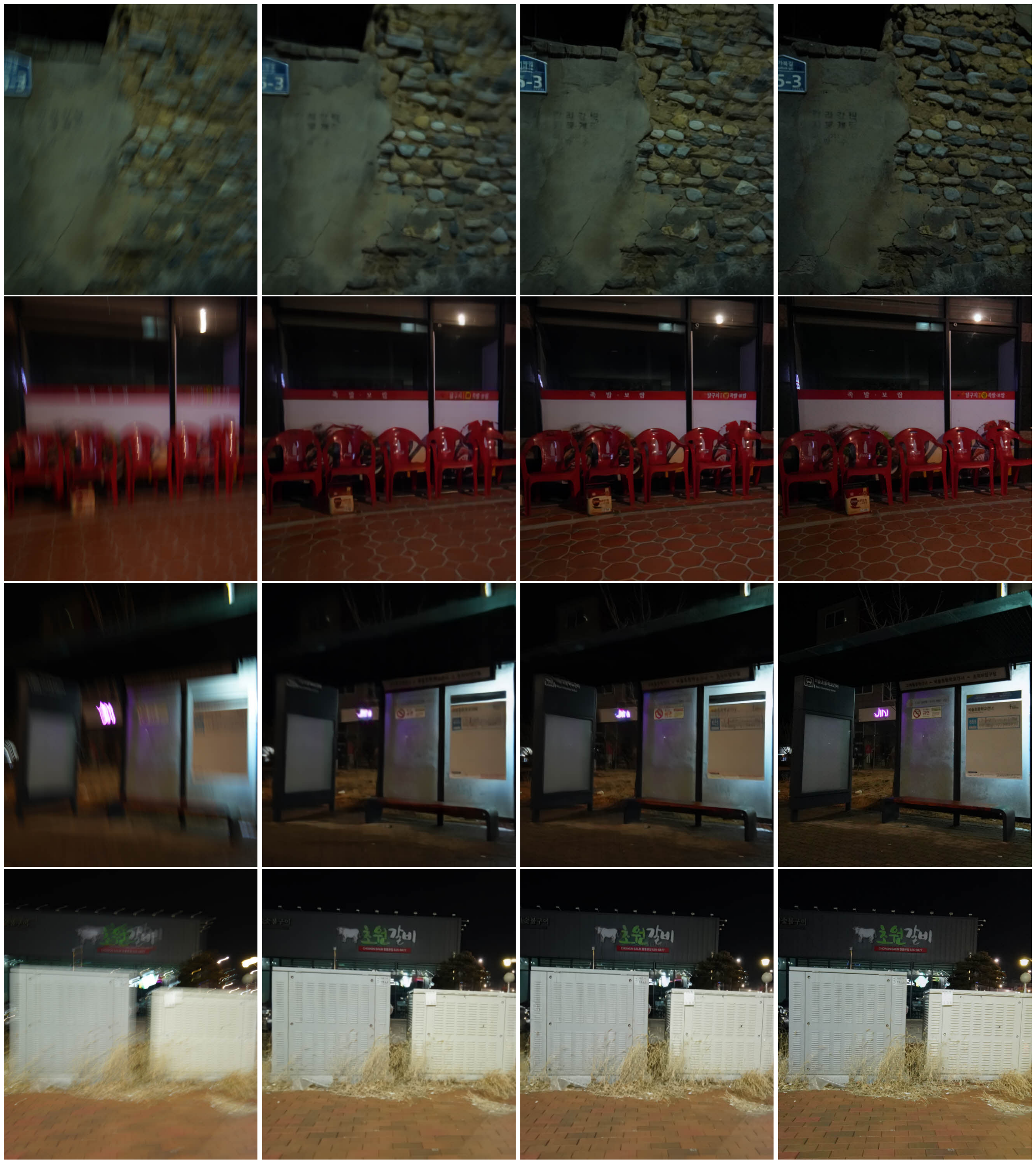

(a) Blur Input (b) MAXIM-3S (c) RefDeblur-B (d) Sharp

Figure 8: Additional qualitative results on RealBlur (Rim et al., 2020).