This CVPR paper is the Open Access version, provided by the Computer Vision Foundation.
Except for this watermark, it is identical to the accepted version;
the final published version of the proceedings is available on IEEE Xplore.

K3DN: Disparity-aware Kernel Estimation for Dual-Pixel Defocus Deblurring

Yan Yang[1] * Liyuan Pan[2] † * Liu Liu[3] MiaoMiao Liu[1]
[1]ANU [2]BITSZ & CSAT, BIT [3]Cyberverse Dept., Huawei
Yan.Yang@anu.edu.au liyuan.pan@bit.edu.cn liuliu33@huawei.com miaomiao.liu@anu.edu.au

Abstract

The dual-pixel (DP) sensor captures a two-view image pair in a single snapshot by splitting each pixel in half. The disparity occurs in defocus blurred regions between the two views of the DP pair, while the in-focus sharp regions have zero disparity. This motivates us to propose a K3DN framework for DP pair deblurring, and it has three modules: i) a disparity-aware deblur module. It estimates a disparity feature map, which is used to query a trainable kernel set to estimate a blur kernel that best describes the spatially-varying blur. The kernel is constrained to be symmetrical per the DP formulation. A simple Fourier transform is performed for deblurring that follows the blur model; ii) a reblurring regularization module. It reuses the blur kernel, performs a simple convolution for reblurring, and regularizes the estimated kernel and disparity feature unsupervisedly, in the training stage; iii) a sharp region preservation module. It identifies in-focus regions that correspond to areas with zero disparity between DP images, aims to avoid the introduction of noises during the deblurring process, and improves image restoration performance. Experiments on four standard DP datasets show that the proposed K3DN outperforms state-of-the-art methods, with fewer parameters and flops at the same time.

1. Introduction

Dual-pixel (DP) sensors are widely employed in digital single-lens reflex cameras (DSLR) and smartphone cameras. Each pixel of a DP sensor is divided into two photodiodes to provide two sub-views of a scene, namely, left and right views, to assist in auto-focusing. Recently, researchers have used the two-view DP pair to benefit several computer vision tasks, especially defocus deblurring [1,2,24]. Specifically, blurred pixels in the left and right view of the DP pair exhibit an amount of shift, termed DP disparity, which provides information for the blur kernel estimation (the key for defocus deblurring [14, 16, 19]).

† Corresponding author. * Equal contribution.

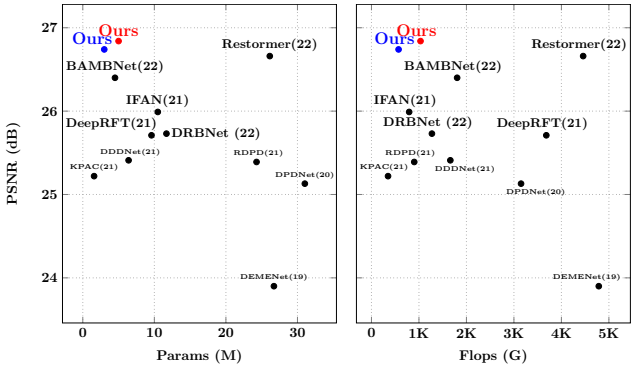


Figure 1. *Comparison of our method and state-of-the-art methods on the real DPD-blur dataset [1]. The Blue and red cycles separately denote our tiny and lightweight models. We show PSNRs (dB) with respect to model parameters (M) and Flops (G). The numbers in the brackets beside methods denote publication years. Best viewed in color on the screen.*

Existing DP-based defocus deblurring methods are generally divided into two categories: end-to-end-based and disparity (defocus map)-based methods. End-to-end based methods [1, 2, 11, 17, 19, 24] directly restore an all-in-focus image from a defocus blurred DP pair without considering DP domain knowledge for network design. Therefore, it is challenging for these methods to tackle spatially-varying and large blur. Disparity-based methods [10, 11, 15, 23] estimate the DP disparity (or defocus map) to aid the restoration of an all-in-focus image. However, these methods either require extra data (ground-truth DP disparity as supervision) [15], neglect DP disparity knowledge when warping [10], heavily depend on the number of network branches [11], or rely on pre-calibrated kernel sets and additional priors [23] for training.

Our method, K3DN, belongs to the disparity-based group, and it has high performance on real DP datasets (see Fig. 1 for an example). K3DN follows the mathematic indications of DP and blur models, while not requiring pre-calibrated kernel sets and extra data. K3DN consists of three modules: i) a disparity-aware deblur module; ii) a reblurring regularization module; iii) a sharp region preservation module.

The first component is a disparity-aware deblur module that estimates spatially-varying kernels for deblurring. We define a trainable kernel set, and kernels in the set can model diverse blur patterns. Each kernel satisfies the horizontally symmetric constraint [2, 16], following the DP image formulation. Given the input DP image pair, we first estimate a disparity feature map, and then use the disparity feature map to query the kernel set to estimate a DP blur kernel that best describes the spatially-varying blur. With the blur kernel, we simply perform a Fourier transform for deblurring that follows the blur model.

The second component we developed is a reblurring regularization module. It reuses the DP blur kernel, takes an all-in-focus image as input, and performs a simple convolution to generate a blurred image. By enforcing the similarity between the blurred image and the input DP image, we regularize the proposed deblur module.

Our third component is a sharp region preservation module. Not all regions of DP pairs are blurred. We observe that previous works neglect to preserve pixel values or features from in-focus sharp regions. By mining (based on the DP model) similarities between features before and after our deblur module, we preserve features from all-in-focus regions and improve feature quality, leading to better image restoration performance.

Please note that all intermediate parts (*e.g.*, the trainable kernel set and disparity estimator) do not require extra ground-truth information (*e.g.*, pre-calibrated kernel sets and disparity) for training. We only need ground-truth all-in-focus images for end-to-end training.

Our contributions are summarized below:

- A simple K3DN framework for DP image pair defocus deblurring;
- A disparity-aware deblur module. It has a trainable kernel set, and estimates a disparity feature map to query the set to estimate a DP blur kernel that best describes the spatial varying blur;
- A reblurring regularization module. It follows the DP model, reuses the blur kernel from the deblur module, and regularizes the deblur module;
- A sharp region preservation module. It mines in-focus regions of DP pairs, improves feature quality from in-focus regions, resulting in better restoration performance.

Experimentally, we validate the proposed K3DN on four standard benchmark datasets [1, 2, 15, 16], showing state-of-the-art performance in image restoration and model size.

2. Related Works

End-to-end-based method. Existing methods directly restore all-in-focus images from defocus blurred DP pairs. Abuolaim *et al*. [1] propose the first deep learning framework for DP pair defocus deblurring. In their following work [2], a synthetic dataset (that uses the Butterworth filter to benefit approximations of DP kernels) is proposed to help model performance. Instead of using the left and right view of the DP pair, several works consider a center view (i. e., an average) of the DP pair. Ruan *et al*. [17] use filter dynamic networks [5] to adaptively infer pixel-wise convolution kernels. Son *et al*. iteratively learn deblur kernels for different feature channels with the shape attention mechanism [26]. Zamir *et al*. [24] introduce the self-attention mechanism [21] over feature channels to this task. Though favourable performance is achieved, they ignore the DP formulation during network design.

Disparity-based method. This strategy restores an all-in-focus image while estimating DP disparity or defocus map. Pan *et al*. [15] use ground-truth DP disparity during network training, but the DP disparity is hard to obtain. Liang *et al*. [11] use the translating disk proposed by Abhijith *et al*. [16] to estimate a four-layer defocus mask by a four-branch network, and then employ another four-branch network for deblurring. Lee *et al*. [10] estimate disparity unsupervisedly by using a spatial transformer to warp a left-view DP image to its right view. However, the warping ignores the difference between the disparity of the classical stereo model and the DP model [16]. Xin *et al*. [23] decompose the blurred image layer-wisely to estimate the defocus map and the all-in-focus image jointly. However, manually defined parameters and pre-calibrated kernel sets are required, which is hard to generalize. Although several works use rebur loss to regularize the deblur model, their rebur models need to be specially trained with extra parameters.

This paper follows disparity-based approaches but is different from the past works in three main factors: i) unlike [10, 15], our blur kernels are shared in the deblurring and reblurring processes through only an inverse operation; ii) different from [11, 23], our K3DN do not need those pre-calibrated kernel set and pre-defined thresholds to layerwise the defocus mask/map; iii) we observe the importance of preserving the non-blurred (i. e., in-focus) regions when deblurring, and preserve those sharp features by following the DP formulation and blur model.

3. K3DN

3.1. Problem Definition

For a DP sensor, the left and right views $(\mathbf{B}_{\mathrm{L}}, \mathbf{B}_{\mathrm{R}})$ are formed by light rays of the scene that pass through the left and right sub-aperture. When a point is outside the depth-of-field (DoF) of the sensor, its reflected rays are defocused, and this point is defocus blurred in the left and right views while forming a relative shift, *e.g.*, points on the surface of the sphere in Fig. 2. For light rays coming from a point inside the DoF, there is no difference between the left and right view images, and they are both in-focus, *e.g.*, points

on the surface of the cone in Fig. 2.

Assuming constant scene depth, the left and right DP images $(\mathbf{B}_{\mathrm{L}}, \mathbf{B}_{\mathrm{R}})$ can be separately modeled by left and right kernels $(\mathbf{K}_{\mathrm{L}}, \mathbf{K}_{\mathrm{R}})$ and an associated all-in-focus image $\mathbf{I}$,

$$\mathbf{B}_{\mathrm{L}} = \mathbf{I} \otimes \mathbf{K}_{\mathrm{L}}, \quad \mathbf{B}_{\mathrm{R}} = \mathbf{I} \otimes \mathbf{K}_{\mathrm{R}}\,, \tag{1}$$

where $\otimes$ denotes the convolution operation. The left kernel equals the right kernel flipped about the vertical axis (which is the axis perpendicular to the axis of disparity), i.e., $\mathbf{K}_{\mathrm{L}} = \mathrm{flip}(\mathbf{K}_{\mathrm{R}})$[1], and vice versa.

For $\mathbf{B}_{\mathrm{L}}$ and $\mathbf{B}_{\mathrm{R}}$, they can be thought of as a two-sample lightfield [16, 23] with a sum equivalent to the image captured by a regular sensor $\mathbf{B}$,

$$\mathbf{B} = \frac{1}{2}(\mathbf{B}_{\mathrm{L}} + \mathbf{B}_{\mathrm{R}}) = \mathbf{I} \otimes \frac{1}{2}(\mathbf{K}_{\mathrm{L}} + \mathbf{K}_{\mathrm{R}}) = \mathbf{I} \otimes \mathbf{K}\,, \tag{2}$$

where $\mathbf{K}$ is horizontally symmetric based on the DP model.

Given a scene with varying depth, the defocus blur in DP images is spatially-varying. In other words, a blurred image is formed by multiple blur kernels. Regions/Layers of the DP pair with the same disparity share a specific blur kernel.

For the all-in-focus image $\mathbf{I}$, it can be decomposed into multiple layers via

$$\mathbf{I} = \int_{t_n}^{t_f} \mathbf{I} \odot \mathbf{M}(t)\, dt, \quad \int_{t_n}^{t_f} \mathbf{M}(t)\, dt = \mathbf{1}, \tag{3}$$

where t is the layer index and corresponds to a specific scene depth (or image disparity). t_n and t_f are the near and far depth bounds, respectively. $\mathbf{M}(t)$ is an impulse function, to sample image pixels corresponding to the t-th layer. $\odot$ denotes the element-wise multiplication, and $\mathbf{1}$ denotes an all-ones matrix with the same size as $\mathbf{I}$.

We numerically evaluate a layer of $\mathbf{I}$ using quadrature,

$$\mathbf{I} = \sum_{i=1}^{T} \mathbf{I}^i, \quad \mathbf{I}^i = \mathbf{I} \odot \mathbf{M}^i, \tag{4}$$

where $\mathbf{M}^i$ is a binary mask with value ones in the region corresponding to the i-th layer of $\mathbf{I}$. T is the number of quantization intervals (layers). Pixels in $\mathbf{I}^i$ have the same disparity.

Substituting Eq. 4 into Eq. 2, we have the DP model,

$$\mathbf{B} = \sum_{i=1}^{T} \mathbf{B}^i = \sum_{i=1}^{T} \mathbf{I}^i \otimes \mathbf{K}^i\,, \tag{5}$$

where $\mathbf{B}^i = \mathbf{I}^i \otimes \mathbf{K}^i$ is the i-th layer of the DP image selected by $\mathbf{M}^i$.

Transforming convolution into element-wise multiplication in the frequency domain, we have

$$\mathcal{F}(\mathbf{B}^i) = \mathcal{F}(\mathbf{I}^i) \odot \mathcal{F}(\mathbf{K}^i)\,, \tag{6}$$

[1]The assumption of symmetric left and right kernels has been widely used in the community [4, 16]. However, a special case occurred in the Pixel phone, where the left and right kernels additionally spatially-varying for constant scene depth. This is caused by the optical imperfections of cheap smartphone lenses [23].

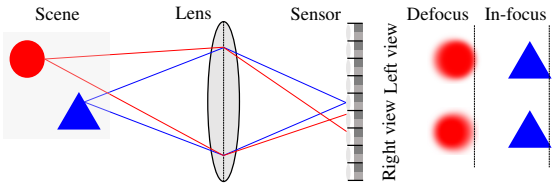


Figure 2. *Illustration of the DP image formulation. Here is an example of a sphere outside and a cone within the camera DoF. Rays from the sphere form two defocus blurred regions on the left/right DP views with a detectable disparity, where the disparity is directly correlated to the amount of defocus blur, i. e., blur kernels. Rays from points of the cone object intersect at the sensor and are in-focused. The resulting regions in the DP pairs (i. e., left and right views) exhibit no disparity. An auxiliary line is drawn to indicate the DP disparity.*

where $\mathcal{F}(\cdot)$ denotes the discrete Fourier transform. The restoration of the all-in-focus image $\mathbf{I}$ is given by

$$\mathbf{I} = \sum_{i=1}^{T} \mathbf{B}^i \otimes \mathcal{F}'\left(\frac{1}{\mathcal{F}(\mathbf{K}^i)}\right)\,, \tag{7}$$

where $\mathcal{F}'(\cdot)$ is the inverse discrete Fourier transform. Note that Eq. 5 and Eq. 7 separately constrain kernels $\mathbf{K}^i$ used in the blurring and deblurring processes in a closed-form expression. Based on this observation, our key idea is to use a neural network to estimate a kernel that best describes the blurring of $\mathbf{I}^i$ and deblurring of $\mathbf{B}^i$.

Specifically, we use a network to estimate a kernel set $\mathcal{K} = \{\mathbf{K}^i \mid i = 1, \cdots, N\}$. For the i-th layer of the blurred image $\mathbf{B}$, we first estimate its corresponding disparity features and then use these features to query the kernel set $\mathcal{K}$, to estimate the best kernel that corresponds to $\mathbf{B}^i$. Finally, the estimated kernel is used to deblur $\mathbf{B}^i$. Collecting all deblurred image layers composes our restored image $\hat{\mathbf{I}}$.

Overall architecture. In practice, considering that neural networks are good at extracting features, we propose to first embed original blurry DP images into a high-dimensional feature space and then perform the proposed deblurring process (Eq. 7) on feature maps channel-wisely. Finally, a decoder network is used to restore $\hat{\mathbf{I}}$ from deblurred feature maps. Our overall framework is given in Fig. 3. The following section details each component of our K3DN.

3.2. Spatial Varying Deblurring

In this section, we elaborate on all components of our K3DN. Detailed component structures and parameters are given in the supplementary material.

Encoder. We use a convolution-based encoder $\mathbf{E}(\cdot, \cdot)$ to embed a DP image pair into a high-dimensional feature space, i. e., extract a feature map $\mathbf{F}_{\mathrm{B}} = \mathbf{E}(\mathbf{B}_{\mathrm{L}}, \mathbf{B}_{\mathrm{R}})$, where $\mathbf{F}_{\mathrm{B}} \in \mathbb{R}^{\mathsf{H_f} \times \mathsf{W_f} \times \mathsf{C_f}}$. $\mathsf{H_f}$, $\mathsf{W_f}$, and $\mathsf{C_f}$ are the height, width, and channel of the encoded feature map, respectively.

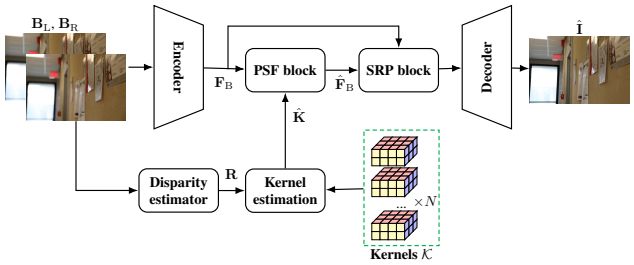


Figure 3. *Overall architecture of K3DN. Given input blurry DP images* $(\mathbf{B}_\mathrm{L}, \mathbf{B}_\mathrm{R})$*, they are simultaneously passed to an encoder to extract a feature map* $\mathbf{F}_\mathrm{B}$*, and a disparity estimator to extract a disparity feature map* $\mathbf{R}$*. Note that* $\mathbf{R}$ *and* $\mathbf{F}_\mathrm{B}$ *are spatially aligned, i. e., for each feature vector* $\mathbf{r}^i \in \mathbf{R}$*, we can easily find its corresponding features* $\mathbf{F}_\mathrm{B}^i$ *(the* i*-th layer of* $\mathbf{F}_\mathrm{B}$*).* $\mathbf{R}$ *is used to query a trainable kernel set* $\mathcal{K}$ *to estimate the best kernel* $\hat{\mathbf{K}}^i$ *for* $\mathbf{F}_\mathrm{B}^i$*. A PSF block takes* $\mathbf{F}_\mathrm{B}$ *and its corresponding spatial-varying kernels as input, and performs a simple Fourier transform to deblur* $\mathbf{F}_\mathrm{B}$*, and outputs a deblurred feature map* $\hat{\mathbf{F}}_\mathrm{B}$*. By comparing the similarity of* $\mathbf{F}_\mathrm{B}$ *and* $\hat{\mathbf{F}}_\mathrm{B}$*, an SRP block mines in-focus sharp regions within DP images, and preserves features from these regions to improve the feature map quality. Finally, a decoder restores sharp image* $\hat{\mathbf{I}}$ *from the output feature map of the SRP block.*

Disparity estimator. We borrow the architecture from [3], and adapt a lightweight encoder $\mathbf{D}(\cdot,\cdot)$ to extract a disparity feature map $\mathbf{R} = \mathbf{D}(\mathbf{B}_\mathrm{L}, \mathbf{B}_\mathrm{R})$, where $\mathbf{R} \in \mathbb{R}^{\mathsf{H_d} \times \mathsf{W_d} \times \mathsf{C_d}}$.

The alignment of encoder and disparity estimator. Note that $\mathbf{F}_\mathrm{B}$ and $\mathbf{R}$ are spatially aligned since we only perform convolution operations on a DP pair. $\mathbf{R} \in \mathbb{R}^{\mathsf{H_d} \times \mathsf{W_d} \times \mathsf{C_d}}$ can be considered as a set of $\mathsf{C_d}$-dimensional feature vectors $\mathbf{r}^i \in \mathbb{R}^{\mathsf{C_d} \times 1}$, which is extracted at $\mathsf{H_d} \times \mathsf{W_d}$ spatial locations, i. e., $\{\mathbf{r}^i | i = 1, ..., \mathsf{H_d} \times \mathsf{W_d}\}$. The corresponding i-th layer of the feature map $\mathbf{F}_\mathrm{B}$ can be found by performing nearest neighbor interpolation [2], and is given by $\mathbf{F}_\mathrm{B}^i \in \mathbb{R}^{\frac{\mathsf{H_f}}{\mathsf{H_d}} \times \frac{\mathsf{W_f}}{\mathsf{W_d}} \times \mathsf{C_f}}$. Note that $\mathbf{F}_\mathrm{B}^i$ with the same disparity feature vector $\mathbf{r}^i$ share a specific blur kernel. $\mathbf{r}^i$ is used as a proxy to query a trainable kernel set for estimating the specific blur kernel.

Kernel estimation. We maintain a trainable kernel set $\mathcal{K} = \{\mathbf{K}^i \mid \mathbf{K}^i \in \mathbb{R}^{\mathsf{H_k} \times \mathsf{W_k} \times \mathsf{C_f}}, i = 1, ..., N\}$ for the feature space, where $\mathsf{H_k}$, $\mathsf{W_k}$, $\mathsf{C_f}$, N are height, width, channel and the number of kernels, respectively. Each $\mathbf{K}^i$ follows Eq. 2 by the symmetry constraint, i.e., $\mathbf{K}_\mathrm{R}^i = \mathrm{flip}(\mathbf{K}_\mathrm{L}^i)$. Note that the channel of kernels equals that of $\mathbf{F}_\mathrm{B}$ since we will estimate a kernel that deblurs $\mathbf{F}_\mathrm{B}$ channel-wisely.

By comparing $\mathbf{r}^i$ and $\mathcal{K}$, we estimate a weighting vector $\mathbf{a}^i \in \mathbb{R}^{N \times 1}$, describing the contribution of each element in the kernel set $\mathcal{K}$. We have

$$\mathbf{a}^i = \sigma\Big(\mathbf{W_v}\big(\mathrm{Sum}(\mathbf{W}_\mathcal{K} \circledast \mathcal{K}) \odot (\mathbf{W_r}\mathbf{r}^i)\big)\Big), \tag{8}$$

$\mathbf{W}_\mathcal{K} \in \mathbb{R}^{\mathsf{H_k} \times \mathsf{W_k} \times \mathsf{C_f} \times N}$ is a weighting tensor to project ($\circledast$) the kernel set. Specifically, for the i-th kernel $\mathbf{K}^i \in \mathcal{K}$, the i-th slice along the last dimension of $\mathbf{W}_\mathcal{K}$ is used to weight (Hadamard product) $\mathbf{K}^i$. $\mathrm{Sum}(\cdot)$ returns the sum of all elements for each weighted kernel and outputs a N-dim vector. $\mathbf{W_r} \in \mathbb{R}^{N \times \mathsf{C_d}}$ and $\mathbf{W_v} \in \mathbb{R}^{N \times N}$ are weighting matrices. $\sigma(\cdot)$ is the Softmax function to enforce $\sum \mathbf{a}^i = 1$.

We use the weighting vector $\mathbf{a}^i$ to aggregate the trainable kernel set $\mathcal{K}$, and estimate a blur kernel $\hat{\mathbf{K}}^i$ describing the blurring process of $\mathbf{F}_\mathrm{B}^i$,

$$\hat{\mathbf{K}}^i = \sum_{j=1}^{N} a_j^i \mathbf{K}^j, \tag{9}$$

where a_j^i is the j-th element of $\mathbf{a}^i$, and it denotes the weight of the j-th trainable kernel $\mathbf{K}^j$.

Eq. 8 and Eq. 9 can be viewed as a differentiable pooling of N trainable kernels. To avoid hard-assignment of one kernel from $\mathcal{K}$ to $\mathbf{F}_\mathrm{B}^i$, which is not differentiable, we replace it with a soft assignment of kernels to $\mathbf{F}_\mathrm{B}^i$, by estimating a $\hat{\mathbf{K}}^i$ using $\mathbf{r}^i$ as a proxy. Practically, we use sparse representation for $\mathcal{K}$ to reduce trainable parameters, keeping only one sub-kernel of $\mathbf{K}^i \in \mathcal{K}$ to be non-zero.

PSF block. With $\hat{\mathbf{K}}^i$ and $\mathbf{F}_\mathrm{B}^i$, the proposed PSF block performs a simple Fourier transform to deblur $\mathbf{F}_\mathrm{B}^i$ channel-wisely. The deblurred i-th feature layer $\hat{\mathbf{F}}_\mathrm{B}^i$ is given by

$$\hat{\mathbf{F}}_\mathrm{B}^i = \mathbf{F}_\mathrm{B}^i \otimes \mathcal{F}'\left(\frac{1}{\mathcal{F}(\hat{\mathbf{K}}^i)}\right), i = 1, ..., \mathsf{H_d} \times \mathsf{W_d}\ . \tag{10}$$

Eq. 10 can be implemented easily via depth-wise convolution in Pytorch or Tensorflow. Collecting all deblurred feature layers composes our deblurred feature map $\mathbf{F}_\mathrm{B}$.

SRP block. Some feature layers $\mathbf{F}_\mathrm{B}^i$ should not be deblurred [3] as they correspond to in-focus regions of DP images. We observe features from these regions retained similarity before and after the PSF block. Therefore, we compare $\mathbf{F}_\mathrm{B}$ and $\hat{\mathbf{F}}_\mathrm{B}$, use an SRP block $\mathbf{S}(\cdot,\cdot)$ to mine in-focus regions, and preserve features from these regions to improve the feature quality. Motivated by the attention mechanism [21], we define the similarity $\mathbf{O}_\mathrm{B}$ between $\mathbf{F}_\mathrm{B}$ and $\hat{\mathbf{F}}_\mathrm{B}$ as

$$\mathbf{O}_\mathrm{B} = \mathrm{Attention}\left(\mathtt{Q}_\mathrm{B}, \mathtt{K}_\mathrm{B}\right), \tag{11}$$

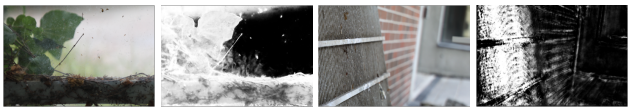
$$\mathtt{Q}_\mathrm{B} = \Phi_\mathtt{Q}\left(\mathrm{CAT}(\mathbf{F}_\mathrm{B}, \hat{\mathbf{F}}_\mathrm{B})\right), \mathtt{Q}_\mathrm{B} \in \mathbb{R}^{\mathsf{H_f} \times \mathsf{W_f} \times 2\mathsf{C_f}}, \tag{12}$$

$$\mathtt{K}_\mathrm{B} = \Phi_\mathtt{K}\left(\mathrm{CAT}(\mathbf{F}_\mathrm{B}, \hat{\mathbf{F}}_\mathrm{B})\right), \mathtt{K}_\mathrm{B} \in \mathbb{R}^{\mathsf{H_f} \times \mathsf{W_f} \times 2\mathsf{C_f}}, \tag{13}$$

where $\mathrm{CAT}(\cdot,\cdot)$ denotes concatenation. $\Phi_\mathtt{Q}(\cdot)$ and $\Phi_\mathtt{K}(\cdot)$ are two simple convolutions to obtain the query $\mathtt{Q}_\mathrm{B}$ and $\mathtt{K}_\mathrm{B}$ from

[2] If the sizes of an image and its disparity map are different, the only correct way to find the disparity value of an image pixel is performing nearest neighbor interpolation, to avoid the wrong interpolation at disparity discontinuity regions/boundaries.

[3] Ideally, deblurring with a impulse function. In practice, we can only approximate it while introducing unwanted artifacts.



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

Figure 4. *(a)(c): Two sample DP images* $\mathbf{B}$*; (b)(d): Visualizations of their corresponding* $\mathbf{O}_{\mathrm{B}}$*. Note that we can easily identify sharp regions from* $\mathbf{O}_{\mathrm{B}}$*, which corresponds to bright regions. Since features from sharp image regions are preserved before and after our deblur module (i. e.,* $\mathbf{F}_{\mathrm{B}}^{i}$ *and* $\hat{\mathbf{F}}_{\mathrm{B}}^{i}$ *are similar), their dot products are large, resulting in bright regions.*

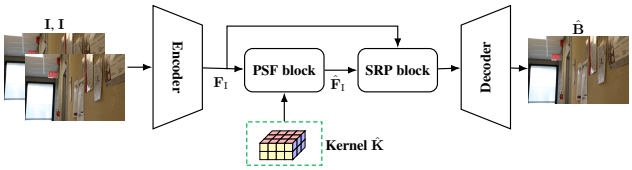


Figure 5. *Overall architecture of our reblurring process. Given input in-focus DP image* $\mathbf{I}$*, it is first duplicated and then passed to the same encoder, PSF block, SRP block, and decoder given in Sec. 3.2. Note that the same kernel* $\hat{\mathbf{K}}$ *(Eq. 9) is used as an input to the PSF block.*

original feature maps. To save computations, we only perform local attention operations. Specifically, for a feature vector of $\mathtt{Q}_{\mathrm{B}}$, it only attends to (dot product) spatially nearby feature vectors in $\mathtt{K}_{\mathrm{B}}$, rather than all features in $\mathtt{K}_{\mathrm{B}}$. We constrain the attention window size at $s \times s$. Aggregating attention scores for all query vectors in $\mathtt{Q}_{\mathrm{B}}$, we obtain the similarity $\mathbf{O}_{\mathrm{B}} \in \mathbb{R}^{\mathsf{H_f} \times \mathsf{W_f} \times \mathsf{s}^2}$. We provide a visualization of $\mathbf{O}_{\mathrm{B}}$ in Fig. 4.

The similarity (or attention score map) $\mathbf{O}_{\mathrm{B}}$ is used to refine features $\mathbf{F}_{\mathrm{B}}$ and $\hat{\mathbf{F}}_{\mathrm{B}}$, and the refined feature map $\mathbf{S}(\mathbf{F}_{\mathrm{B}}, \hat{\mathbf{F}}_{\mathrm{B}})$ is given by

$$\mathbf{S}(\mathbf{F}_{\mathrm{B}}, \hat{\mathbf{F}}_{\mathrm{B}}) = \mathrm{FFN}\left(\mathtt{V}_{\mathrm{B}} + \mathtt{V}_{\hat{\mathrm{B}}}\right), \tag{14}$$

$$\mathtt{V}_{\mathrm{B}} = \Phi_{\mathtt{V}_1}\left(\mathbf{O}_{\mathrm{B}}\right) \odot \mathbf{F}_{\mathrm{B}}, \mathtt{V}_{\mathrm{B}} \in \mathbb{R}^{\mathsf{H_f} \times \mathsf{W_f} \times \mathsf{C_f}}, \tag{15}$$

$$\mathtt{V}_{\hat{\mathrm{B}}} = \Phi_{\mathtt{V}_2}\left(\mathbf{O}_{\mathrm{B}}\right) \odot \mathbf{F}_{\hat{\mathrm{B}}}, \mathtt{V}_{\hat{\mathrm{B}}} \in \mathbb{R}^{\mathsf{H_f} \times \mathsf{W_f} \times \mathsf{C_f}}, \tag{16}$$

where $\mathtt{V}_{\mathrm{B}}$ and $\mathtt{V}_{\hat{\mathrm{B}}}$ are the aggregated values from $\mathbf{F}_{\mathrm{B}}$ and $\hat{\mathbf{F}}_{\mathrm{B}}$, respectively. $\mathrm{FFN}(\cdot)$ is a feed forward network.

Decoder. Given refined feature map $\mathbf{S}(\mathbf{F}_{\mathrm{B}}, \hat{\mathbf{F}}_{\mathrm{B}})$ from the SRP block, We use a convolution based decorder $\mathbf{G}(\cdot)$ to restore an all-in-focus image $\hat{\mathbf{I}}$.

3.3. Regularization by Reblurring

We have presented our deblurring process in Sec. 3.2. The key finding is that we can estimate a kernel $\hat{\mathbf{K}}^{i}$ describing the blurring process of $\mathbf{F}_{\mathrm{B}}^{i}$. According to Eq. 7 and Eq. 6, the deblurring and blurring processes use the same kernel. In this section, we present our reblurring (blurring) process to regularize the learning of $\hat{\mathbf{K}}^{i}$. Please note that our reblurring process is only used in the network training stage. The overall framework is given in Fig. 5.

In the training stage, we use the provided in-focus DP image $\mathbf{I}$ as an input to our reblurring process, and restore a blurry image $\hat{\mathbf{B}} = \frac{1}{2}(\hat{\mathbf{B}}_{\mathrm{L}} + \hat{\mathbf{B}}_{\mathrm{R}})$. Note that the clear feature map $\mathbf{F}_{\mathrm{I}}$ is blurred by the kernel $\hat{\mathbf{K}}$, to generate a blurry feature map $\hat{\mathbf{F}}_{\mathrm{I}}$,

$$\hat{\mathbf{F}}_{\mathrm{I}}^{i} = \mathbf{F}_{\mathrm{I}}^{i} \otimes \hat{\mathbf{K}}^{i}, \qquad i = 1, ..., \mathsf{H_d} \times \mathsf{W_d}. \tag{17}$$

The blurry feature map $\hat{\mathbf{F}}_{\mathrm{I}}$ is further refined by the SRP block and decoded to restore $\hat{\mathbf{B}}$. By supervising $\hat{\mathbf{B}}$ using the ground-truth blurry image $\mathbf{B}$, we regularize the learning of the blur kernel $\hat{\mathbf{K}}$.

Remark. Note that [10] and [23] also have reblur schemes. However, their reblur schemes use independent parameters, thus their reblurring and blurring processes use different kernels. This limitation reduces their models' generalization ability. Please refer to comparisons on the DPD-disp dataset (Fig. 7) and our supplementary material.

3.4. Loss

To train our network, we have deblurring loss $\mathcal{L}_{deb}$ and reblurring loss $\mathcal{L}_{reb}$ (refer to our supplementary material for details). Our overall training loss $\mathcal{L}$ is given by

$$\mathcal{L} = \mathcal{L}_{\mathrm{deb}}(\mathbf{I}, \hat{\mathbf{I}}) + \lambda_1 \mathcal{L}_{\mathrm{reb}}(\mathbf{B}, \hat{\mathbf{B}}), \tag{18}$$

where λ_1 is a balancing weight.

4. Experiment

4.1. Experimental setup

Synthetic dataset. We evaluate our method on two synthetic DP datasets, namely, the DDD-syn [15] and RDPD [2] datasets. DDD-syn provides 5,000 and 500 pairs of defocus and all-in-focus images for training and testing, respectively. This dataset is synthesized from the NYU RGBD dataset [18], approximating the DP kernel with integral image. RDPD provides 10,115 training and 1,005 testing image pairs that are synthesized from the SYNTHIA dataset [6] by using the Butterworth filter.

Real dataset. We evaluate our method on two real DP datasets, namely, the DPD-blur [1] and DPD-disp [16] datasets. They are captured by the Canon EOS 5D Mark IV. DPD-blur provides 350 training and 76 testing image pairs. DPD-disp has 100 defocus blurred DP images, without providing corresponding all-in-focus images. Our model is qualitatively evaluated on the DPD-disp dataset, using pretrained model on the DPD-blur dataset.

Evaluation metric. We evaluate the quality of restored images with using standard metrics, i. e., peak signal-to-noise ratio (PSNR), structural similarity (SSIM) [22], relative error (RMSE_rel) [15], and mean absolute error (MAE). Moreover, we use the number of parameters and flops (with

Table 1. *Comparison of image restoration performance and computation cost on the DPD-blur dataset [1]. '*' indicates the method is trained with extra data. We highlight the best and the second-best numbers in red and blue, respectively.*

Method	Restoration Performance				Computation Cost	
	PSNR↑	SSIM↑	RMSE_rel$_{(10^{-2})}$ ↓	MAE$_{(10^{-1})}$ ↓	Params (M)	Flops (G)
Input	23.89	0.725	6.39	0.471	-	-
EBDB [7]	23.69	0.707	6.54	0.471	-	-
DMENet [9]	23.90	0.720	6.38	0.470	26.71	4787
DPDNet [1]	25.13	0.787	5.54	0.401	31.03	3150
KPAC [19]	25.22	0.774	5.48	0.400	1.58	349
RDPD* [2]	25.39	0.772	5.38	0.400	24.28	901
DDDNet [15]	25.41	0.786	5.36	0.385	6.40	1661
DRBNet* [17]	25.73	0.791	5.17	0.393	11.69	1273
DeepRFT [14]	25.71	0.801	5.18	0.389	9.60	3682
IFAN [10]	25.99	0.804	5.01	0.373	10.48	794
BAMBNet [11]	26.40	0.821	4.79	0.450	4.50	1804
Restormer [24]	26.66	0.833	4.65	0.350	26.13	4458
Ours (Tiny)	26.74	0.825	4.60	0.351	3.00	571
Ours (Lightweight)	26.84	0.829	4.55	0.349	5.00	1033
Ours (Large)	27.06	0.835	4.44	0.341	15.90	3352

Table 2. *Comparison of image restoration performance on the DDD-syn dataset [15].*

Method	PSNR↑	SSIM↑	RMSE_rel$_{(10^{-2})}$ ↓	MAE$_{(10^{-1})}$ ↓
Input	31.06	0.895	2.80	0.165
EBDB [7]	26.48	0.683	4.74	0.363
DMENet [9]	30.14	0.939	3.11	0.177
DPDNet [1]	31.45	0.926	2.68	0.160
RDPD* [2]	32.42	0.912	2.39	0.155
DDDNet [15]	33.21	0.956	2.17	0.094
DeepRFT [14]	36.53	0.952	1.49	0.096
IFAN [10]	34.18	0.929	1.95	0.124
BAMBNet [11]	35.90	0.954	1.60	0.117
Restormer [24]	36.44	0.957	1.50	0.104
Ours (Tiny)	36.18	0.953	1.55	0.113
Ours (Lightweight)	37.50	0.960	1.33	0.083
Ours (Large)	38.23	0.965	1.23	0.076

Table 3. *Comparison of image restoration performance on the RDPD dataset [2].*

Method	PSNR↑	SSIM↑	RMSE_rel$_{(10^{-2})}$ ↓	MAE$_{(10^{-1})}$ ↓
Input	25.32	0.679	5.42	0.313
EBDB [7]	24.23	0.637	6.14	0.357
DMENet [9]	25.17	0.731	5.51	0.319
DPDNet [1]	29.84	0.828	3.22	0.250
RDPD* [2]	31.09	0.861	2.79	0.160
DDDNet [15]	30.14	0.840	3.11	0.231
DeepRFT [14]	31.71	0.879	2.60	0.162
IFAN [10]	31.12	0.865	2.78	0.181
BAMBNet [11]	31.78	0.867	2.58	0.169
Restormer [24]	32.27	0.871	2.43	0.164
Ours (Tiny)	31.82	0.887	2.56	0.168
Ours (Lightweight)	32.22	0.894	2.45	0.151
Ours (Large)	32.84	0.905	2.28	0.142

image size at $1120 \times 1680 \times 3$) to measure the computation cost of each method.

State-of-the-art method. Our method is compared with EBDB [7], DMENet [9], DPDNet [1], KPAC [19], DDDNet [15], RDPD [2], DRBNet [17], DeepRFT [14], IFAN [10], BAMBNet [11], and Restormer [24].

Implementation detail. Our network is trained from scratch using the AdamW optimizer [12] with a learning rate of 3×10^{-4} and a batch size of 4. Please refer to the supplementary material for adapting our model on single image defocus deblurring tasks.

4.2. Results

Quantitative evaluation. The comparisons with state-of-the-art methods on the DPD-blur, DDD-syn, and RDPD datasets are given in Tab. 1, Tab. 2 and Tab. 3, respectively. For our method, we train three models (Tiny, Lightweight, and Large) by varying the number of parameters (see our supplementary material for details). We have the following observations according to the comparisons: i) with a similar number of parameters and flops as the second-best method Restormer, our method (Large) consistently outperforms all methods on all datasets. For example, the PSNR (dB) of our method (Large) and Restormer on the DPD-blur, DDD-syn, and RDPD datasets are 27.06/26.66, 38.23/36.44, 32.84/32.27, respectively; ii) though the performance of the second-best method Restormer is comparable with respect to our lightweight model, Restormer has large computation costs (26.13M parameters and 4458G flops). In contrast, our lightweight model (5.00M parameters and 1033G flops) has small computation costs, and ranks third place overall.

Qualitative evaluation. The comparisons with state-of-the-art methods on the two real datasets DPD-blur and DPD-disp are given in Fig. 6 and Fig. 7, respectively. The results show that our method restores clearer images with sharp edges than state-of-the-art methods. For example, our restored texts in Fig. 6 and Fig. 7 are clearer than others.

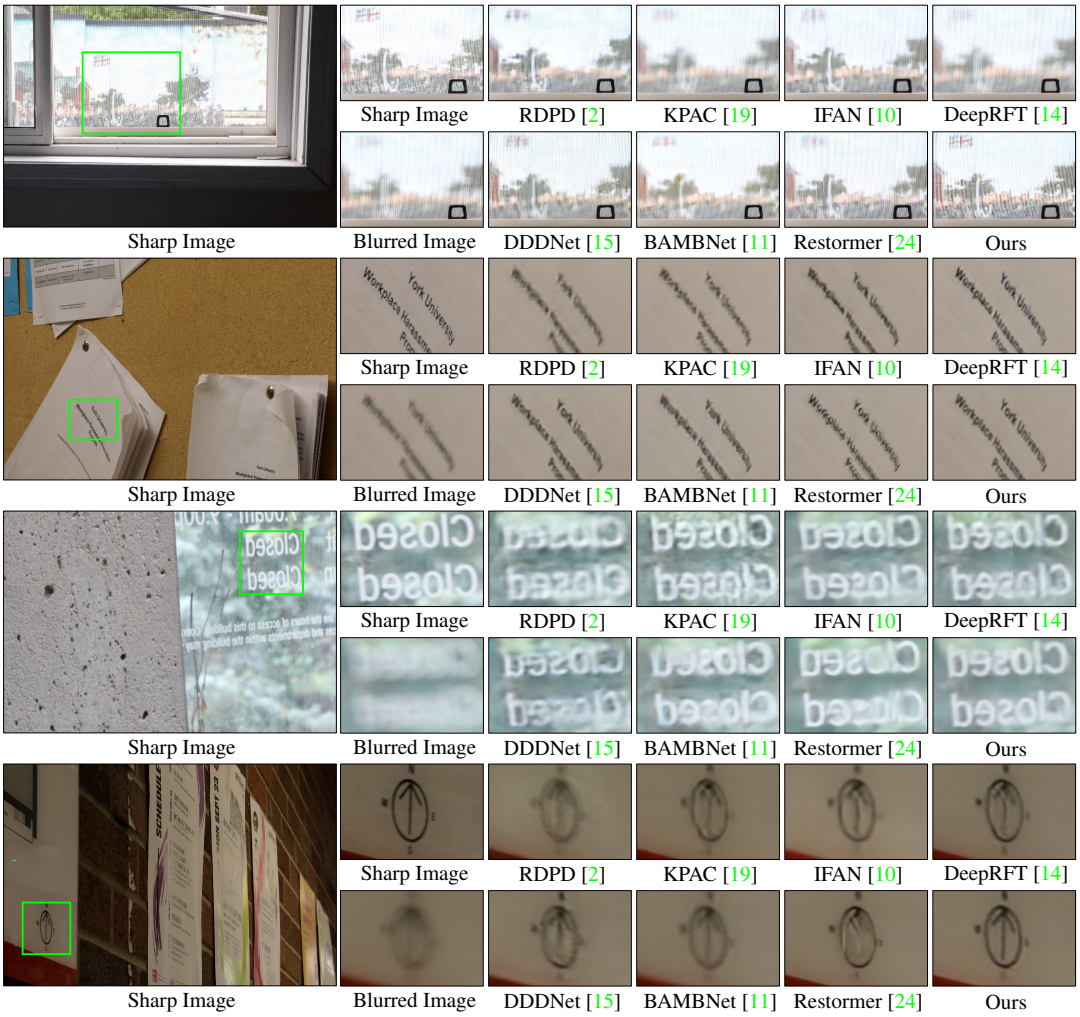


Figure 6. *Comparison of image restoration performance on the DPD-blur dataset [1]. The large sharp images in the first column are ground-truth sharp images. The small sharp images in the second column are cropped images from the green bounding box in the large ground-truth sharp images. The blurred images in the second column are corresponding input blurry images ($\mathbf{B}_{\mathrm{L}}$).*

Table 4. *The effectiveness of our network modules.*

Components			PSNR↑	SSIM↑
PSF Block	SRP Block	Reblur		
✓	✓	✓	26.84	0.829
✓	✓	✗	26.80	0.828
✓	✗	✓	26.76	0.826
✗	✓	✗	26.69	0.825
✓	✗	✗	26.68	0.823

Sharp region preservation. Not all regions in an input blurry DP image are defocus blurred, and pixel values of these sharp (or in-focus) regions should not be altered by a deblurring method. To show the efficacy of our method in preserving sharp regions of an input DP image, we manually label sharp regions of testing DP images of the real DPD-blur dataset. We compare pixel values of input DP images and output restored clear images, corresponding to these sharp regions. We use two metrics: i) an error map, i. e., absolute pixel value difference; ii) intensity error distribution, i. e., the percentage of pixels with respect to different pixel value difference thresholds. One sample error map and the intensity error distribution are given in Fig. 8. We have the following observation. Most current state-of-the-art methods largely alter pixel values of in-focus sharp regions. In contrast, our method can detect and preserve pixel values of in-focus sharp regions, and outperforms current state-of-the-art methods.

4.3. Ablation

Architecture. We first validate the effectiveness of our network modules, i. e., the PSF block, SRP block, and Reblurring regularization. The results are given in Tab. 4. It shows that the best performance is obtained when all three network modules are used. Removing modules will reduce

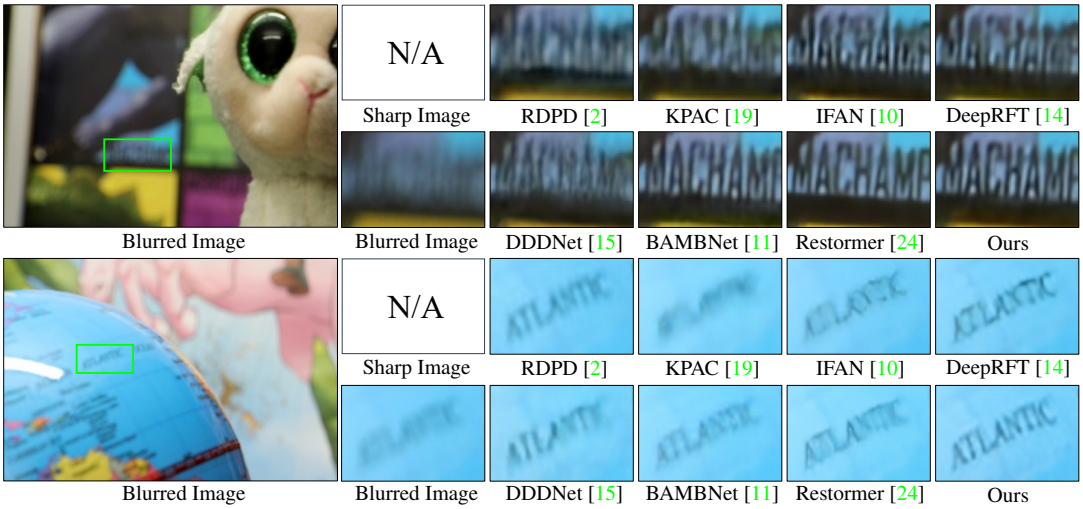


Figure 7. *Comparison of image restoration performance on the DPD-disp dataset [16]. The large blurred images in the first column are input blurry images ($\mathbf{B}_{\mathrm{L}}$). The small blurred images in the second column are cropped images from the green bounding box in the large input blurry images. Note that the DPD-disp dataset does not provide ground-truth sharp images.*

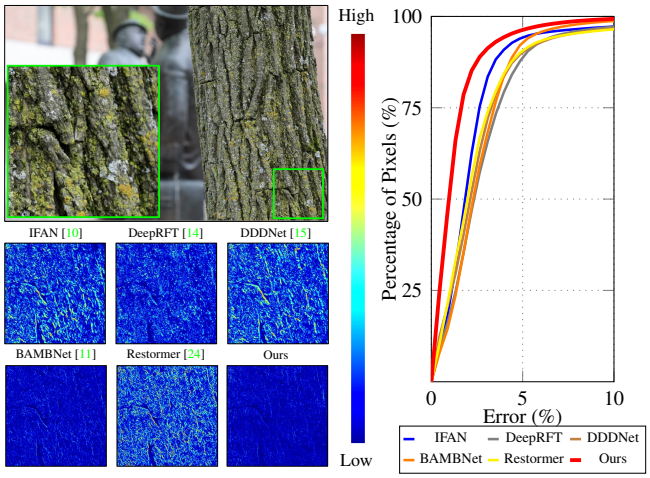


Figure 8. *The efficacy of sharp region preservation. Left: error maps of different methods. The brighter the color, the larger the error. Right: Intensity error (i. e., 'RMSE_rel') distributions of different methods on DPD-blur dataset [1]. We show the percentage of pixels with respect to different error thresholds. At each error threshold, the larger the percentage, the better the method.*

the performance of our model.

Kernel set size. Our method has a trainable kernel set $\mathcal{K}$, with size at N. We study the impact of kernel set size N, by varying N from 64 to 192. The results are given in Tab. 5. It shows that our image restoration performance improves with respect to an increasing number of kernels. We set $N = 128$, balancing the computational cost and performance of our model.

Scalability. We investigate the scalability of our method by building three model variants (i. e., tiny, lightweight, and large) of K3DN. They have a different number of parameters and flops (please refer to supplementary material for details). The comparisons of the model performance on the DPD-blur, DDD-syn, and RDPD datasets are given in Tab. 1, Tab. 2 and Tab. 3, respectively. The results show that our large model achieves the best performance. Please note that the PSNR of our tiny model is better than those of current state-of-the-art methods on the real DPD-blur dataset.

Table 5. *The impact of the size of kernel set $\mathcal{K}$.*

N	64	96	128	160	192
PSNR↑	26.71	26.76	26.84	26.85	26.87
SSIM↑	0.822	0.823	0.829	0.829	0.830

5. Conclusion

In this paper, we have proposed the K3DN framework for DP pair defocus deblurring. It has three new modules. We first maintain a trainable kernel set and use DP disparity to dynamically estimate a kernel that best describes the spatial varying blur. With the estimated kernel, our PSF block performs a simple Fourier transform for deblurring. We then use an SRP block to preserve in-focus sharp regions of the input DP images. Finally, in the training stage, we propose a reblurring module to regularize the estimated kernel, by performing a simple convolution. Experimental results on both synthetic and real datasets show that our method outperforms state-of-the-art methods.

Acknowledgment. This research was supported in part by the Beijing Institute of Technology Research Fund Program for Young Scholars, and MiaoMiao Liu was supported by the ARC fellowship and Discovery Project grant (DE180100628, DP200102274).

References

[1] Abdullah Abuolaim and Michael S Brown. Defocus deblurring using dual-pixel data. In *European Conference on Computer Vision*, pages 111–126. Springer, 2020. 1, 2, 5, 6, 7, 8, 11, 12, 15, 16, 17, 18, 19

[2] Abdullah Abuolaim, Mauricio Delbracio, Damien Kelly, Michael S Brown, and Peyman Milanfar. Learning to reduce defocus blur by realistically modeling dual-pixel data. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 2289–2298, 2021. 1, 2, 5, 6, 7, 8, 11, 15, 16, 17, 18, 19

[3] Xuelian Cheng, Yiran Zhong, Mehrtash Harandi, Yuchao Dai, Xiaojun Chang, Hongdong Li, Tom Drummond, and Zongyuan Ge. Hierarchical neural architecture search for deep stereo matching. In Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin, editors, *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020. 4

[4] Rahul Garg, Neal Wadhwa, Sameer Ansari, and Jonathan T. Barron. Learning single camera depth estimation using dual-pixels. In *2019 IEEE/CVF International Conference on Computer Vision, ICCV 2019, Seoul, Korea (South), October 27 - November 2, 2019*, pages 7627–7636. IEEE, 2019. 3

[5] Xu Jia, Bert De Brabandere, Tinne Tuytelaars, and Luc Van Gool. Dynamic filter networks. In Daniel D. Lee, Masashi Sugiyama, Ulrike von Luxburg, Isabelle Guyon, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain*, pages 667–675, 2016. 2

[6] Daniel Hernández Juárez, Lukas Schneider, Antonio Espinosa, Juan C. Moure, David Vázquez, Antonio M. López, Uwe Franke, and Marc Pollefeys. Slanted stixels: Representing san francisco's steepest streets. In *British Machine Vision Conference 2017, BMVC 2017, London, UK, September 4-7, 2017*. BMVA Press, 2017. 5

[7] Ali Karaali and Claudio Rosito Jung. Edge-based defocus blur estimation with adaptive scale selection. *IEEE Transactions on Image Processing*, 27(3):1126–1137, 2017. 6

[8] Diederik P. Kingma and Prafulla Dhariwal. Glow: Generative flow with invertible 1x1 convolutions. In Samy Bengio, Hanna M. Wallach, Hugo Larochelle, Kristen Grauman, Nicolò Cesa-Bianchi, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, pages 10236–10245, 2018. 14

[9] Junyong Lee, Sungkil Lee, Sunghyun Cho, and Seungyong Lee. Deep defocus map estimation using domain adaptation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 12222–12230, 2019. 6

[10] Junyong Lee, Hyeongseok Son, Jaesung Rim, Sunghyun Cho, and Seungyong Lee. Iterative filter adaptive network for single image defocus deblurring. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2021, virtual, June 19-25, 2021*, pages 2034–2042. Computer Vision Foundation / IEEE, 2021. 1, 2, 5, 6, 7, 8, 11, 12, 15, 16, 17, 18, 19

[11] Pengwei Liang, Junjun Jiang, Xianming Liu, and Jiayi Ma. Bambnet: A blur-aware multi-branch network for defocus deblurring. *CoRR*, abs/2105.14766, 2021. 1, 2, 6, 7, 8, 11, 15, 16, 17, 18, 19

[12] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019. 6

[13] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019. 11

[14] Xintian Mao, Yiming Liu, Wei Shen, Qingli Li, and Yan Wang. Deep residual fourier transformation for single image deblurring. *CoRR*, abs/2111.11745, 2021. 1, 6, 7, 8, 11, 15, 16, 17, 18, 19

[15] Liyuan Pan, Shah Chowdhury, Richard Hartley, Miaomiao Liu, Hongguang Zhang, and Hongdong Li. Dual pixel exploration: Simultaneous depth estimation and image restoration. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4340–4349, June 2021. 1, 2, 5, 6, 7, 8, 11, 15, 16, 17, 18, 19

[16] Abhijith Punnappurath, Abdullah Abuolaim, Mahmoud Afifi, and Michael S. Brown. Modeling defocus-disparity in dual-pixel sensors. In *2020 IEEE International Conference on Computational Photography, ICCP 2020, Saint Louis, MO, USA, April 24-26, 2020*, pages 1–12. IEEE, 2020. 1, 2, 3, 5, 8, 12

[17] Lingyan Ruan, Bin Chen, Jizhou Li, and Miu-Ling Lam. Learning to deblur using light field generated and real defocus images. *CoRR*, abs/2204.00367, 2022. 1, 2, 6

[18] Nathan Silberman, Derek Hoiem, Pushmeet Kohli, and Rob Fergus. Indoor segmentation and support inference from RGBD images. In *European Conference on Computer Vision*, pages 746–760. Springer, 2012. 5

[19] Hyeongseok Son, Junyong Lee, Sunghyun Cho, and Seungyong Lee. Single image defocus deblurring using kernel-sharing parallel atrous convolutions. In *2021 IEEE/CVF International Conference on Computer Vision, ICCV 2021, Montreal, QC, Canada, October 10-17, 2021*, pages 2622–2630. IEEE, 2021. 1, 6, 7, 8, 11, 15, 16, 17, 18, 19

[20] Mingxing Tan and Quoc V. Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, volume 97 of *Proceedings of Machine Learning Research*, pages 6105–6114. PMLR, 2019. 14

[21] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. 06 2017. 2, 4

[22] Zhou Wang, Alan C. Bovik, Hamid R. Sheikh, and Eero P. Simoncelli. Image quality assessment: from error visibil-

ity to structural similarity. *IEEE Trans. Image Process.*, 13(4):600–612, 2004. 5

[23] Shumian Xin, Neal Wadhwa, Tianfan Xue, Jonathan T. Barron, Pratul P. Srinivasan, Jiawen Chen, Ioannis Gkioulekas, and Rahul Garg. Defocus map estimation and deblurring from a single dual-pixel image. In *2021 IEEE/CVF International Conference on Computer Vision, ICCV 2021, Montreal, QC, Canada, October 10-17, 2021*, pages 2208–2218. IEEE, 2021. 1, 2, 3, 5, 11, 12

[24] Syed Waqas Zamir, Aditya Arora, Salman H. Khan, Munawar Hayat, Fahad Shahbaz Khan, and Ming-Hsuan Yang. Restormer: Efficient transformer for high-resolution image restoration. *CoRR*, abs/2111.09881, 2021. 1, 2, 6, 7, 8, 11, 12, 13, 15, 16, 17, 18, 19

[25] Syed Waqas Zamir, Aditya Arora, Salman H. Khan, Munawar Hayat, Fahad Shahbaz Khan, Ming-Hsuan Yang, and Ling Shao. Multi-stage progressive image restoration. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2021, virtual, June 19-25, 2021*, pages 14821–14831. Computer Vision Foundation / IEEE, 2021. 11

[26] Yulun Zhang, Kunpeng Li, Kai Li, Lichen Wang, Bineng Zhong, and Yun Fu. Image super-resolution using very deep residual channel attention networks. In Vittorio Ferrari, Martial Hebert, Cristian Sminchisescu, and Yair Weiss, editors, *Computer Vision - ECCV 2018 - 15th European Conference, Munich, Germany, September 8-14, 2018, Proceedings, Part VII*, volume 11211 of *Lecture Notes in Computer Science*, pages 294–310. Springer, 2018. 2

[27] Changyin Zhou, Stephen Lin, and Shree Nayar. Coded aperture pairs for depth from defocus. pages 325–332, 09 2009. 12