Single Image Defocus Deblurring Using Kernel-Sharing Parallel Atrous Convolutions: Supplementary Material

Hyeongseok Son Junyong Lee Sunghyun Cho Seungyong Lee
POSTECH
{sonhs, junyonglee, s.cho, leesy}@postech.ac.kr

In this supplementary material, we first present the details of the network architecture (Sec. 1) and then provide additional analyses and experiments: a proof of Eq. (4) in the main paper (Sec. 2), more discussion on kernel upsampling (Sec. 3), more validation examples of Eq. (5) in the main paper (Sec. 4), more examples of inverse kernel sampling using dilated kernels (Sec. 5), blur detection using a scale attention map (Sec. 6), sensitivity to noise (Sec. 7), handling irregular blur (Sec. 8), additional results (Sec. 9), and our model extended to use dual-pixel images (Sec. 10).

1. Detailed Network Architecture

Detailed architectures of overall deblurring network and KPAC block can be found in Tables 1 and 2, respectively.

layer type(#)	size	stride	out	act.
Encoder (3-level)				
*Conv*1_1	5×5	(1, 1)	48	*lrelu*
*Conv*1_2	3×3	(1, 1)	48	*lrelu*
*Conv*2_1	3×3	(2, 2)	48	*lrelu*
*Conv*2_2	3×3	(1, 1)	48	*lrelu*
*Conv*3_1	3×3	(2, 2)	96	*lrelu*
*Conv*3_2	3×3	(1, 1)	96	*lrelu*
*Conv*4_1	3×3	(2, 2)	96	*lrelu*
*Conv*4_2	3×3	(1, 1)	96	*lrelu*
KPAC blocks				
*KPAC*1	5×5	(1, 1)	96	*lrelu*
*KPAC*2	5×5	(1, 1)	96	*lrelu*
concat	*Conv*4_2, *KPAC*1, *KPAC*2			
Decoder (3-level)				
*Conv*5_1	3×3	(1, 1)	96	*lrelu*
*Conv*5_2	3×3	(1, 1)	96	*lrelu*
*Deconv*1	4×4	(2, 2)	96	*lrelu*
cocnat	*Deconv*1, *Conv*3_2			
*Conv*6	3×3	(1, 1)	96	*lrelu*
*Deconv*2	4×4	(2, 2)	48	*lrelu*
cocnat	*Deconv*2, *Conv*2_2			
*Conv*7	3×3	(1, 1)	48	*lrelu*
*Deconv*3	4×4	(2, 2)	48	*lrelu*
cocnat	*Deconv*3, *Conv*1_2			
*Conv*8	5×5	(1, 1)	3	*lrelu*
add	*Conv*8, *Input*			-

Table 1. Architecture of our deblurring network.

2. Proof of $(\frac{1}{s^2}k_{\uparrow s})^{\dagger} = \frac{1}{s^2}(k^{\dagger}{}_{\uparrow s})$

In this section, we present a formal discussion on Eq. (4) in the main paper. For a 2D image l of size $w \times h$, the spatial upsampling can be performed by zero padding to the discrete Fourier transform of l [6, 2]. Let $\uparrow s$ denote the upsampling operation by a scaling factor s, and let $l_{\uparrow s}$ be the upsampled result of image l by the scaling factor s. Then, the discrete Fourier transform $L_{\uparrow s}$ of $l_{\uparrow s}$ is defined in the range $-\frac{sw}{2} \leq u < \frac{sw}{2}, -\frac{sh}{2} \leq v < \frac{sh}{2}$, and can be obtained as:

$$L(u,v)_{\uparrow s} = \begin{cases} L(u,v) & -\frac{w}{2} \leq u < \frac{w}{2}, -\frac{h}{2} \leq v < \frac{h}{2} \\ 0 & \text{otherwise,} \end{cases} \tag{1}$$

where $L(u,v)$ is the discrete Fourier transform of l, and u and v are pixel indices in the frequency domain. $L(0,0)$ corresponds to the DC component of L. This zero padding-based upsampling is mathematically equivalent to convolution with a sinc kernel [2].

In the remaining of this section, for notational simplicity, we use $\uparrow s$ to indicate the zero-padding operation for upsampling in the frequency domain as well as the upsampling operation in the spatial domain. We also omit the pixel coordinates (u,v), e.g., representing $L(u,v)_{\uparrow s}$ as $L_{\uparrow s}$.

We use the Wiener deconvolution [7] to compute the inverse kernel $k^{\dagger}$ of a blur kernel k, *i.e.*,

$$k^{\dagger} = F^{-1}\left(\frac{\overline{F(k)}}{|F(k)|^2 + \epsilon}\right), \tag{2}$$

where $F(k)$ is the discrete Fourier transform of k, and $\overline{F(k)}$ is the complex conjugate of $F(k)$. The division operation is done in an element-wise manner. ϵ is a noise parameter. Note that when $\epsilon = 0$, this inverse kernel is equivalent to the direct inverse kernel $F^{-1}(1/F(k))$.

We first prove the equality relation:

$$(k_{\uparrow s})^{\dagger} = k^{\dagger}{}_{\uparrow s}. \tag{3}$$

Regarding the left side of Eq. (3), the upsampling of a blur kernel k can be obtained using the zero padding-based upsampling as:

$$k_{\uparrow s} = F^{-1}(F(k)_{\uparrow s}). \tag{4}$$

Then, in the frequency domain, from Eq. (1), we have

$$F(k_{\uparrow s}) = F(k)_{\uparrow s} = \begin{cases} F(k) & -\frac{w}{2} \le u < \frac{w}{2}, -\frac{h}{2} \le v < \frac{h}{2} \\ 0 & \text{otherwise.} \end{cases} \tag{5}$$

From Eq. (2), the inverse kernel $(k_{\uparrow s})^{\dagger}$ for the upsampled kernel $k_{\uparrow s}$ can be derived by

$$(k_{\uparrow s})^{\dagger} = F^{-1}\left(\frac{\overline{F(k_{\uparrow s})}}{|F(k_{\uparrow s})|^2 + \epsilon}\right). \tag{6}$$

In the frequency domain, from Eq. (5), we have

$$\begin{aligned} F((k_{\uparrow s})^{\dagger}) &= \frac{\overline{F(k_{\uparrow s})}}{|F(k_{\uparrow s})|^2 + \epsilon} \\ &= \begin{cases} \frac{\overline{F(k)}}{|F(k)|^2 + \epsilon} & -\frac{w}{2} \le u < \frac{w}{2}, -\frac{h}{2} \le v < \frac{h}{2} \\ 0 & \text{otherwise.} \end{cases} \end{aligned} \tag{7}$$

Note that the zero padded area in $F(k_{\uparrow s})$ remains zero in $F((k_{\uparrow s})^{\dagger})$.

Regarding the right side of Eq. (3), from Eqs. (4) and (2), the upsampled inverse kernel $k^{\dagger}{}_{\uparrow s}$ for kernel k can be derived as

$$\begin{aligned} k^{\dagger}{}_{\uparrow s} &= F^{-1}(F(k^{\dagger})_{\uparrow s}) \\ &= F^{-1}\left(\left(F\left(F^{-1}\left(\frac{\overline{F(k)}}{|F(k)|^2 + \epsilon}\right)\right)\right)_{\uparrow s}\right) \\ &= F^{-1}\left(\left(\frac{\overline{F(k)}}{|F(k)|^2 + \epsilon}\right)_{\uparrow s}\right). \end{aligned} \tag{8}$$

Then, in the frequency domain, from Eq. (1), we have

layer type(#)	size	dilation	out	act.
Scale attention module				
AC	5×5	(2, 2)	32	*lrelu*
AC	5×5	(2, 2)	32	*lrelu*
AC	5×5	(2, 2)	16	*lrelu*
AC	5×5	(2, 2)	16	*lrelu*
Conv $\{\alpha_1, ..., \alpha_5\}$	5×5	(1, 1)	5	*sigmoid*
Shape attention module				
global average pooling			96	-
fully connected layer			16	*lrelu*
fully connected layer (β)			48	*sigmoid*
Multiple atrous convolutions				
*AC*1	5×5	(1, 1)	48	*lrelu*
*AC*2	5×5	(2, 2)	48	*lrelu*
*AC*3	5×5	(3, 3)	48	*lrelu*
*AC*4	5×5	(4, 4)	48	*lrelu*
*AC*5	5×5	(5, 5)	48	*lrelu*
Fusion				
cocnat	$\alpha_1 \times \beta \times AC1, ..., \alpha_5 \times \beta \times AC5$			
Conv	3×3	(1, 1)	96	*lrelu*

Table 2. Architecture of our KPAC block. AC denotes an atrous convolution layer.

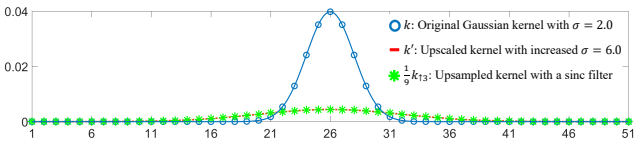
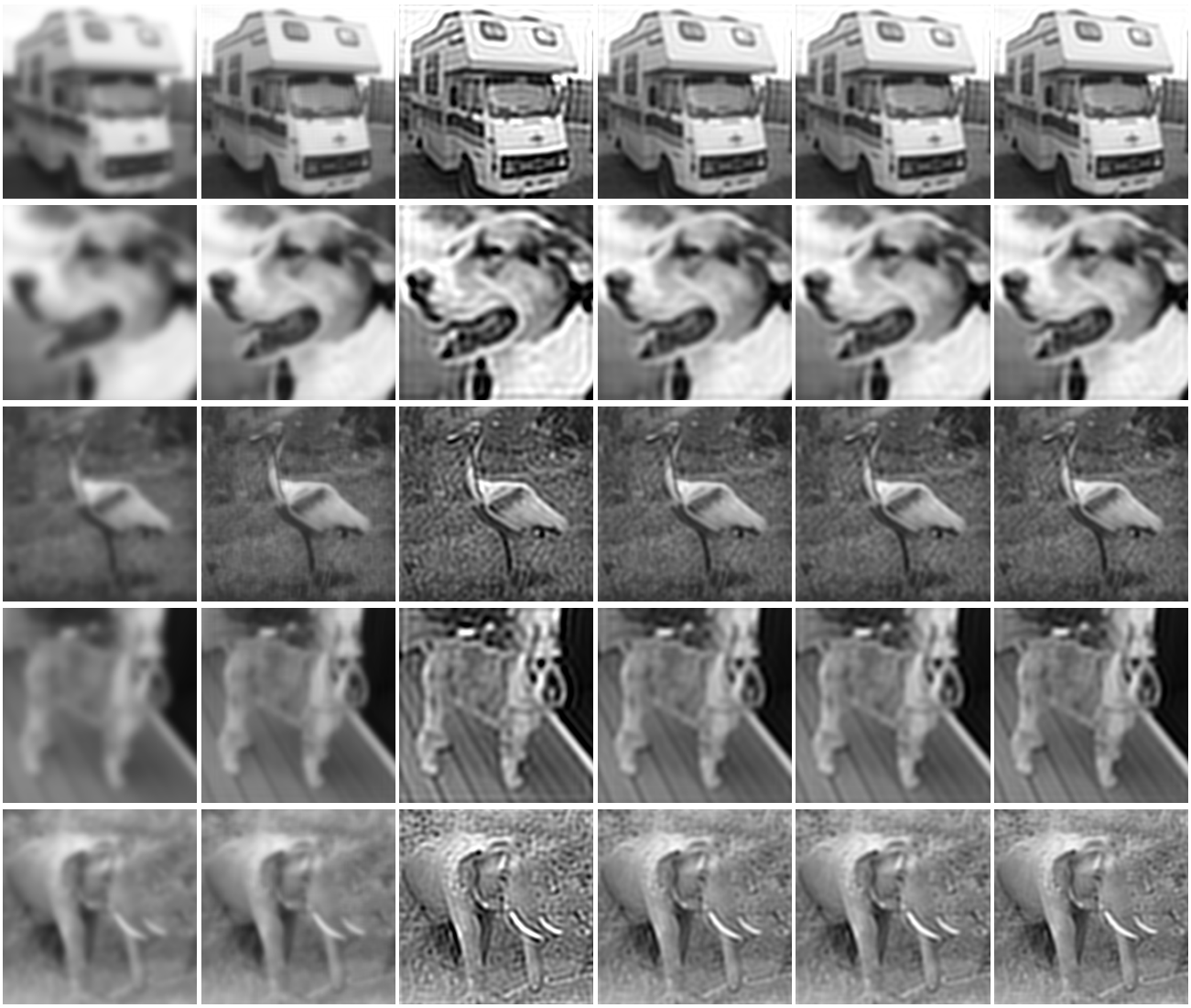


Figure 1. 1D plot of $3\times$ upscaling of a 2D Gaussian kernel. Upsampling ($\frac{1}{9}k_{\uparrow 3}$) the original Gaussian kernel (k) is the same as upscaling (k') the kernel by increasing the standard deviation (σ).

$$\begin{aligned} F(k^{\dagger}{}_{\uparrow s}) &= \left(\frac{\overline{F(k)}}{|F(k)|^2 + \epsilon}\right)_{\uparrow s} \\ &= \begin{cases} \frac{\overline{F(k)}}{|F(k)|^2 + \epsilon} & -\frac{w}{2} \le u < \frac{w}{2}, -\frac{h}{2} \le v < \frac{h}{2} \\ 0 & \text{otherwise.} \end{cases} \end{aligned} \tag{9}$$

As Eqs. (7) and (9) are equivalent to each other, their spatial domain counterparts $(k_{\uparrow s})^{\dagger}$ and $k^{\dagger}{}_{\uparrow s}$ are equivalent too. This proves Eq. (3).

Eq. (4) in the main paper has a scaling factor in both left and right sides. The upsampling operation in the left side of Eq. (3) scales up the total intensity of kernel k by s^2 times. Similarly, the upsampling operation in the right side of Eq. (3) also scales up the total intensity of inverse kernel $k^{\dagger}$ by s^2 times. Thus, to obtain a properly normalized inverse kernel in the left and right sides, we apply a scaling factor $\frac{1}{s^2}$ to $k_{\uparrow s}$ in the left side, and to $k^{\dagger}{}_{\uparrow s}$ in the right side. Then, we obtain Eq. (4) in the main paper.



(a) blurred by $\frac{1}{s_t^2} k_{\uparrow s_t}$ (b) deblurred by $\frac{1}{s_1^2}(k^\dagger_{\uparrow s_1})$ (c) deblurred by $\frac{1}{s_2^2}(k^\dagger_{\uparrow s_2})$ (d) blended using Eq. (5) (e) blended using modified Eq. (5) (f) deblurred by $\frac{1}{s_t^2}(k^\dagger_{\uparrow s_t})$

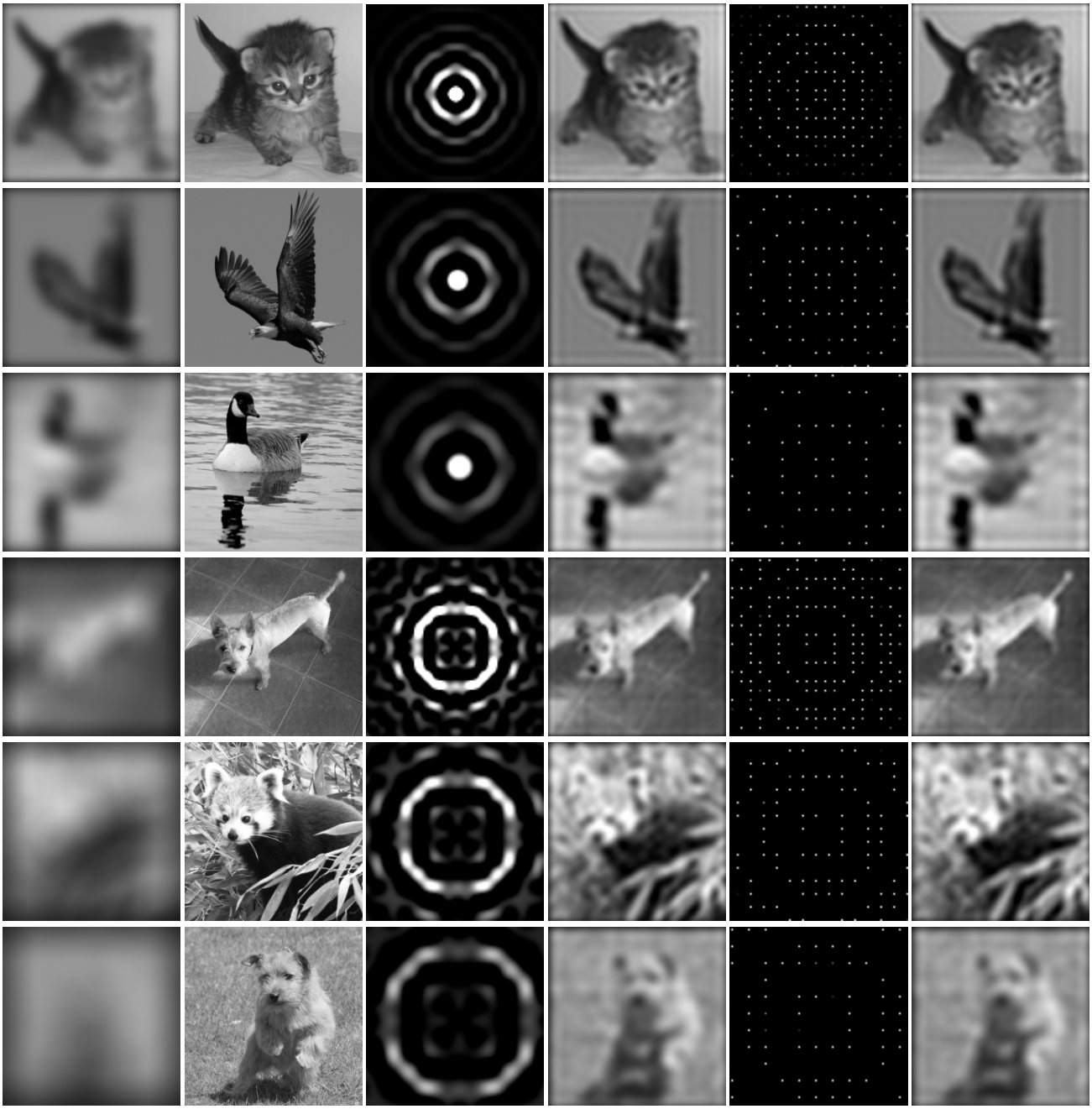
Figure 2. Deconvolution results using Eq. (5) in the main paper and its modified version using dilated inverse kernels. In the examples, s_1 and s_2 are the sampled scale factors and s_t is the target scale factor, where $s_1 < s_t < s_2$. From top to bottom, we used (3.0, 3.4, 4.0), (3.0, 3.5, 4.0), (3.0, 3.6, 4.0), (2.0, 2.6, 3.0), and (1.0, 1.7, 2.0) for the scale factors (s_1, s_t, s_2). We found the blending weights for producing (d) and (e) using non-negative least squares to fit the result of the target scale factor s_t in (f) using those of the sampled scale factors s_1 and s_2 in (b) and (c). Approximation accuracies[1] of (d) from top to bottom are 98.68%, 97.84%, 98.96%, 95.37%, and 97.03%, respectively. Approximation accuracies of (e) from top to bottom are 98.55%, 97.87%, 99.10%, 96.12%, and 97.64%, respectively.

3. Discussion on Kernel Upsampling

We consider general upsampling operation in Eq. 4 in the main paper for the commutative property between upsampling and inversion of a kernel. For validating the property, in Sec. 2 of this supplementary material, we used a specific upsampling method using the sinc filter to change the spatial scale of a kernel. Then, there could be a concern whether upsampling of a blur kernel can model actual scale changes of the blur kernel. In our observation, for Gaussian blur kernels with different standard deviations, which is an often-used assumption in existing defocus deblurring approaches, upsampling a kernel is the same as changing the standard deviation of the kernel (Fig. 1).

Nonetheless, the upsampling method may cause a gap in accurately modeling the scale changes of real-world blur when the blur kernel is arbitrary, other than Gaussian. This potential modeling gap in upsampling would be handled by

[1]The accuracy is computed by $1 - MAE(1, \hat{x}/x_s)$, where MAE is the mean absolute error, $/$ is pixel-wise division, x_s is the deconvolution result using an inverse kernel of a target scale (*e.g.*, Fig. 2f), and $\hat{x}$ is the approximated deconvolution result computed using Eq. (5) (*e.g.*, Fig. 2d).



(a) blurred by $\frac{1}{s^2}k_{\uparrow s}$ (b) ground-truth (c) $\frac{1}{s^2}(k^{\dagger}{}_{\uparrow s})$ (d) deblurred by (c) (e) $k^{\dagger}{}_{\uparrow/s}$ (f) deblurred by (e)

Figure 3. Visual examples showing that $\frac{1}{s^2}(k^{\dagger}{}_{\uparrow s})$ and $k^{\dagger}{}_{\uparrow/s}$ produce similar results. From top to bottom, we used 3, 4, 5, 3, 4, and 5 for the scale factor s. All PSNR values between (d) and (f) are over 40dB.

shape and scale attentions in our KPAC blocks together with other convolution layers in our network.

4. Inverse Kernel-based Deconvolution for Spatially Varying Defocus Blur

In Eq. (5) in the main paper, we approximate spatially varying image deconvolution by combining the results obtained from inverse kernels with a discrete set of sizes. In this section, we present additional visual examples to show the validity of our approximated deconvolution. Figs. 2b and 2c are the deblurred results obtained by convolving inverse kernels of different sizes. While neither kernel fits the actual blur size, a linear combination of the deconvolution results still produces a visually pleasing result (Fig. 2d), almost equivalent to the deconvolution result (Fig. 2f) using the inverse kernel of the target scale.

(a) input (b) EBDB [4] (c) our attention map (d) ground-truth

Figure 4. Blur detection using our scale attention map. The attention map used for blur detection is for the atrous convolution layer with a dilation rate 1 in the first KPAC block of our model.

5. Inverse Kernel Sampling for Atrous Convolution

In Sec. 4.1 of the main paper, we claimed that for a blurred region, which is spatially smooth, filtering operations using sparsely sampled pixels (with a dilated kernel) and using densely sampled pixels (with a rescaled kernel) produce similar results. Fig. 3 presents additional visual examples to show that a dense sampling of an inverse kernel can be replaced by its sparse sampling in terms of deblurring performance.

Moreover, we also claimed that a modified version of Eq. (5) in the main paper using the dilated inverse kernels produces results almost equivalent to those from the original Eq. (5). In Fig. 2, we present additional examples to show the validity of the claim, where the modified and original versions of Eq. (5) produce almost same results in Figs. 2e and 2d, respectively.

6. Blur Detection Using a Scale Attention Map

In Sec. 5.1 of the main paper, we showed that the scale attention map for the atrous convolution layer with a dilation rate of 1 in the first KPAC block of our deblurring network captures blur of almost any size. In this section, we evaluate the blur detection performance of the scale attention map using the CUHK dataset [5]. We use 200 test images in the CUHK dataset and measure F-measure and accuracy. Since our attention is computed in the low resolution feature space, small blurs that can be removed in the encoder network would be ignored. Therefore, we upsample input images four times for obtaining attention maps in this test. The result shows that our attention map (F-measure: 0.832 and accuracy: 78.4%) can detect blur comparably to the recent defocus map estimation method [4] (F-measure: 0.839 and accuracy: 76.5%). Fig. 4 shows qualitative examples.

	Noise level (σ) of testset		
	1%	3%	5%
w/o noise augmentation	25.13	24.76	24.23
w/ noise augmentation	25.06	25.01	24.85

Table 3. Performance (PSNR) under different noise conditions.

7. Sensitivity to Noise

We investigate the sensitivity of our model to different noise levels, as the shape of a desirable inverse kernel can be affected by noise. Table 3 quantitatively shows the effect of the noise level on our model. For the experiment, we prepare two models. One model is trained with the original dataset of [1] (top row of the table). The other model is trained with defocused images augmented with additive Gaussian noise controlled by σ within a range $[0\%, 3\%]$ (bottom row of the table). Compared to the model trained without the noise augmentation, the model trained with the noise augmentation is more robust to noise, and shows more consistent PSNRs around 25 dB.

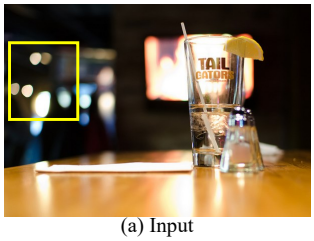
8. Handling Irregular Blur

Due to the network design based on kernel weight sharing, our method would be more effective for the case where the majority of blur variation happens in the size. In practice, blur shape can spatially vary as well, e.g., due to lens distortion in a smartphone camera. Still, we observed that our network moderately works on defocused images captured by smartphones in an unseen dataset [3], which usually contain small-sized blurs (Fig. 5). However, as we discussed as limitations in Sec. 6 of the main paper, our network may not properly handle blur with severely irregular shapes or strong highlights (Fig. 6), which are rarely included in the training set [1].

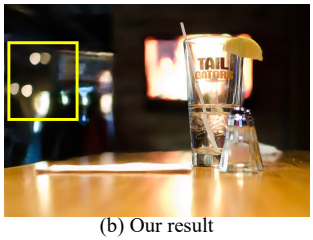
(a) Defocused image captured by a smartphone

(b) Our result

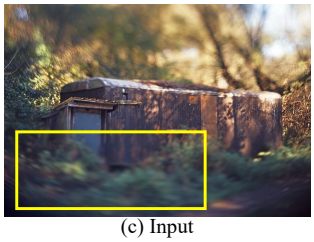
Figure 5. Deblurring on the smartphone dataset [3]. Our network properly removes the blur both in center (yellow box) and peripheral (blue box) regions of an image.



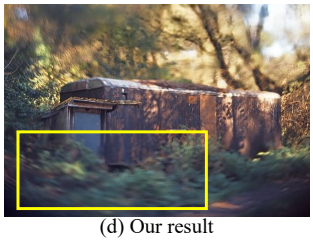
(a) Input



(b) Our result



(c) Input



(d) Our result

Figure 6. Failure cases on the CUHK dataset [5]. Our network fails for irregular blurs such as bokeh with sharp boundaries (a) and swirly bokeh (c), not contained in the training dataset [1].

9. Additional Results

We present additional qualitative results on the DPDD dataset [1] (Figs. 7 and 8) and the CUHK blur detection dataset [5] (Figs. 9 and 10).

10. Our Model Using Dual-Pixel Images

While our model with a single image input shows state-of-the-art performance, the deblurring performance can further boosted by using dual-pixel images as the input. For the experiment, we retrained our model by replacing a single image input as dual-pixel image input with the same training strategy in Sec. 5 of the main paper. Specifically, we concatenate the two images of a dual-pixel image in the channel dimension and use it as the input of the network.

Model	PSNR↑	SSIM↑	LPIPS↓	Parameters (M)
DPDNet (single) [1]	24.42	0.827	0.277	32.25
DPDNet (dual) [1]	25.12	0.850	0.223	32.25
Ours (single)	25.24	0.842	0.225	2.06
Ours (dual)	**25.86**	**0.859**	**0.185**	2.06

Table 4. Quantitative performance of our dual-pixel-based model.

Table 4 shows that dual-pixel input further improves the performance of our model, and our model with dual-pixel input outperforms DPDNet [1] with dual-pixel input by a large margin. Fig. 11 shows that dual-pixel input enables our model to handle fine details better.

References

[1] A. Abuolaim and M.S. Brown. Defocus deblurring using dual-pixel data. In *Proc. ECCV*, 2020. 5, 6, 7, 8, 9, 10, 11

[2] Hiroshi Ashikaga, Heidi Estner, Daniel Herzka, Elliot Mcveigh, and Henry Halperin. Quantitative assessment of single-image super-resolution in myocardial scar imaging. *IEEE Journal of Translational Engineering in Health and Medicine*, 2:1–12, 01 2014. 1

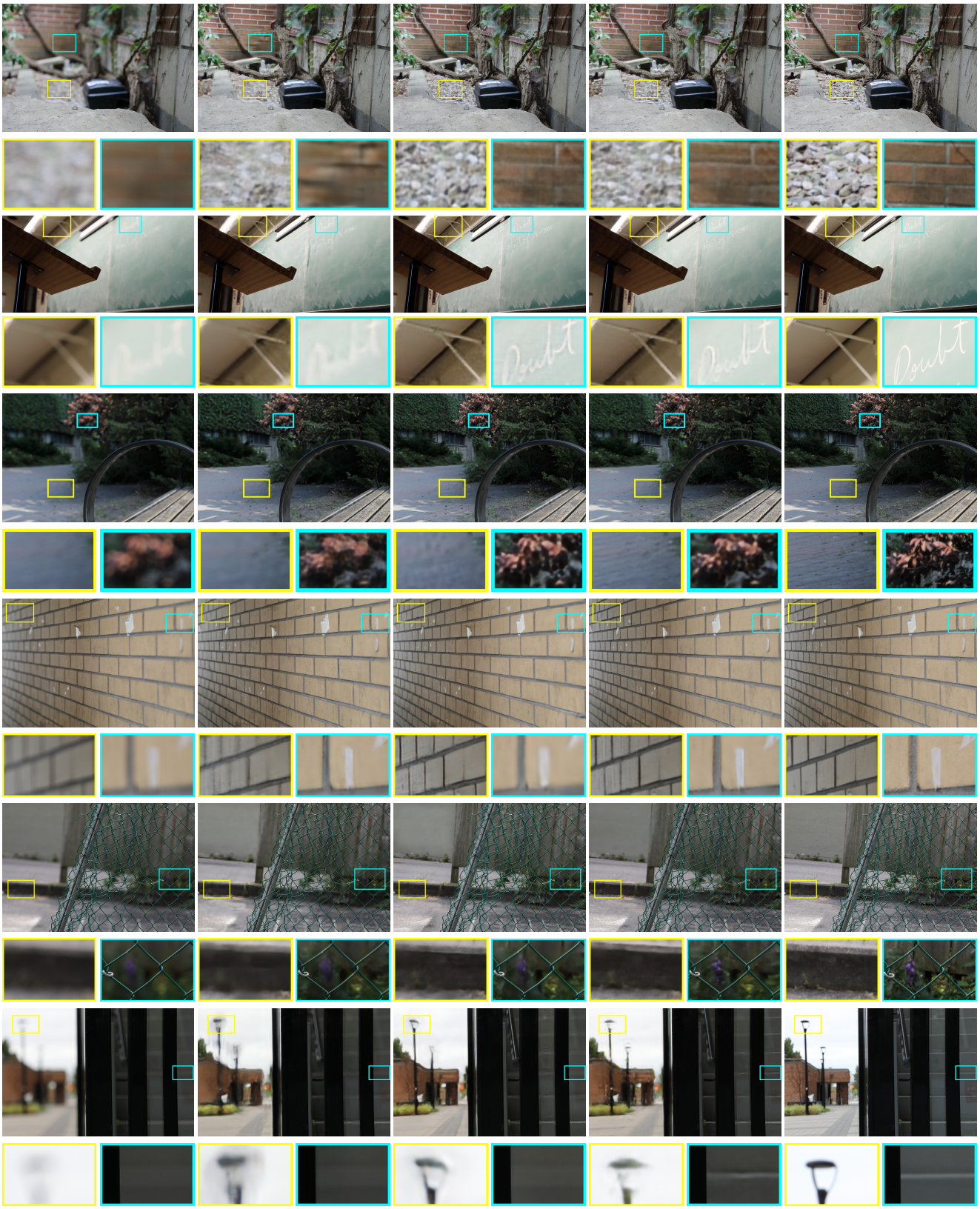
[3] Rahul Garg, Neal Wadhwa, Sameer Ansari, and Jonathan T. Barron. Learning single camera depth estimation using dual-pixels. In *Proc. ICCV*, 2019. 5, 6

[4] A. Karaali and C. Jung. Edge-based defocus blur estimation with adaptive scale selection. *IEEE Trans. Image Processing (TIP)*, 27(3):1126–1137, 2018. 5

[5] J. Shi, L. Xu, and J. Jia. Discriminative blur detection features. In *Proc. CVPR*, 2014. 5, 6, 9, 10

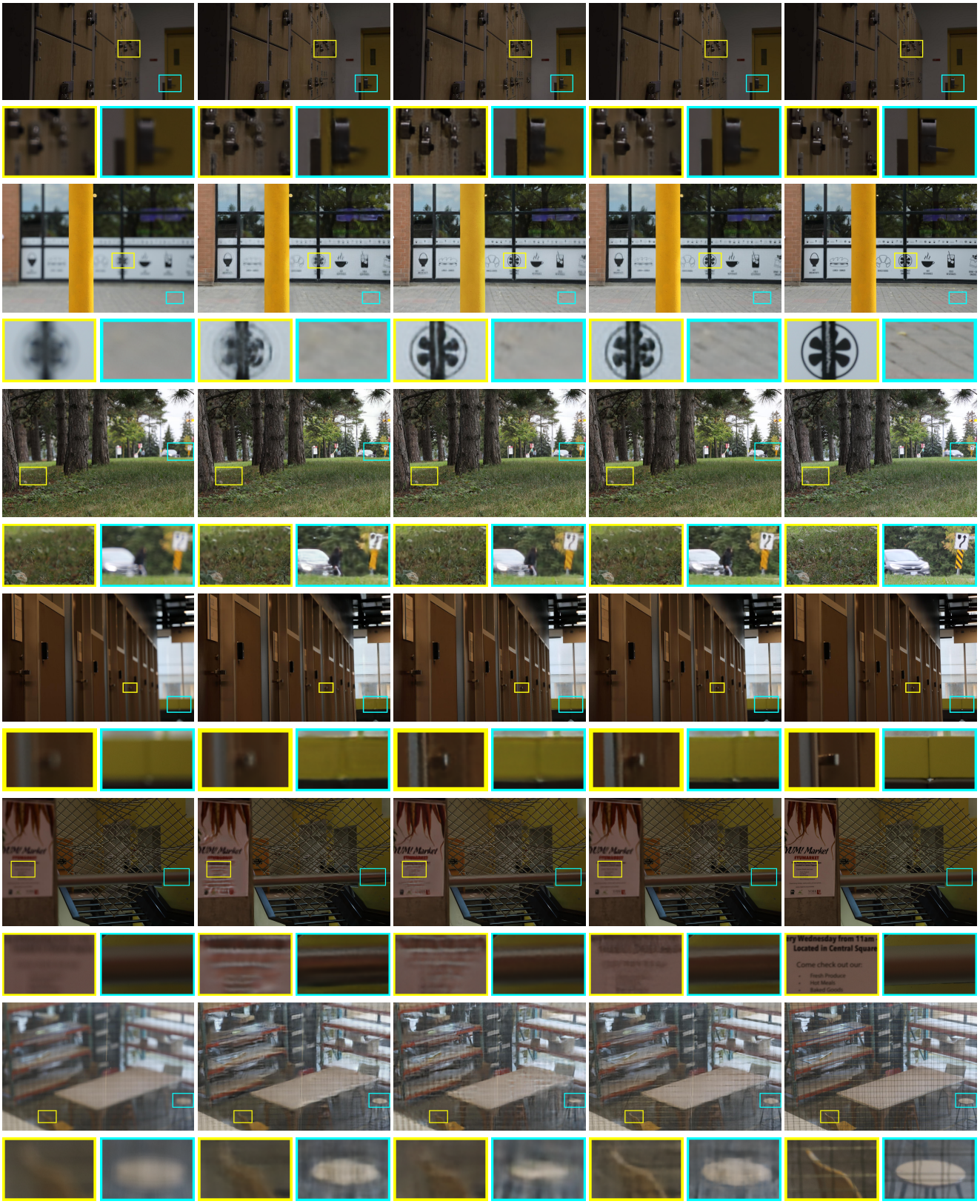
[6] J. O. Smith. *Mathematics of the Discrete Fourier Transform (DFT), with Audio Applications*. W3K Publishing, second edition, 2007. 1

[7] Norbert Wiener. *Extrapolation, Interpolation, and Smoothing of Stationary Time Series*. The MIT Press, 1964. 1



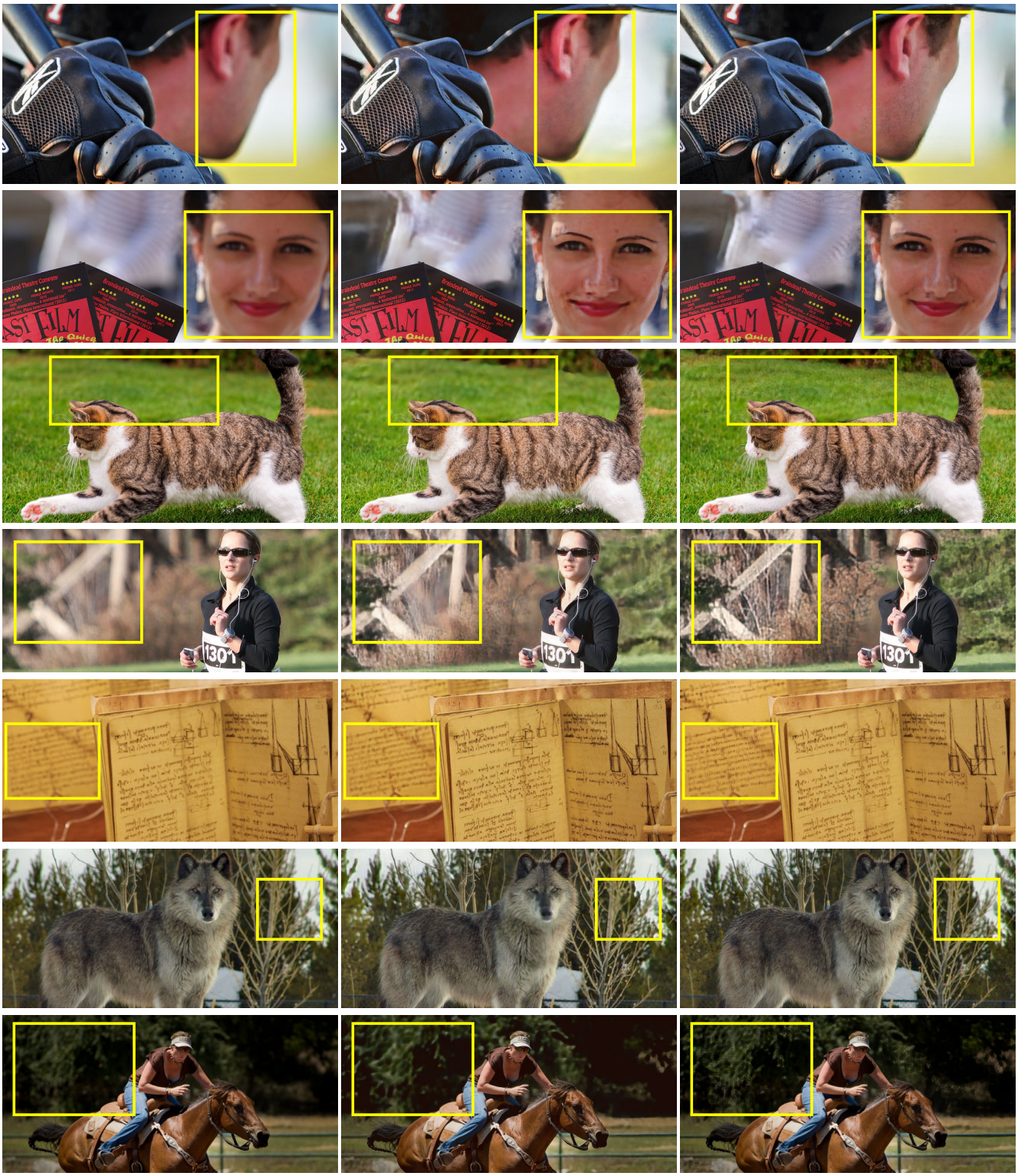
(a) input (b) DPDNet (single) [1] (c) DPDNet (dual) [1] (d) ours (e) GT

Figure 7. Additional qualitative comparisons with DPDNet [1] on the test set of the DPDD dataset [1].



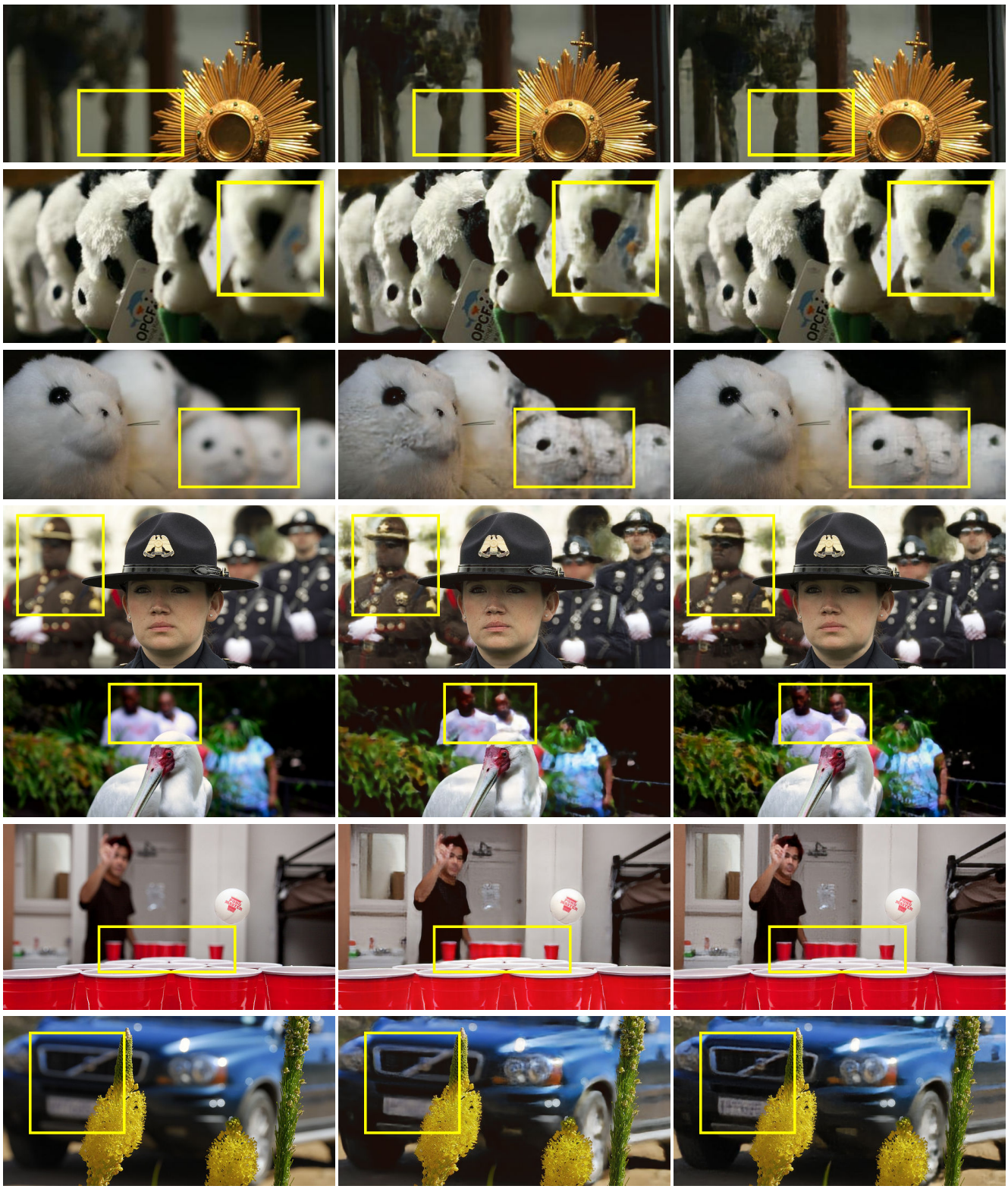
(a) input (b) DPDNet (single) [1] (c) DPDNet (dual) [1] (d) ours (e) GT

Figure 8. Additional qualitative comparisons with DPDNet [1] on the test set of the DPDD dataset [1].



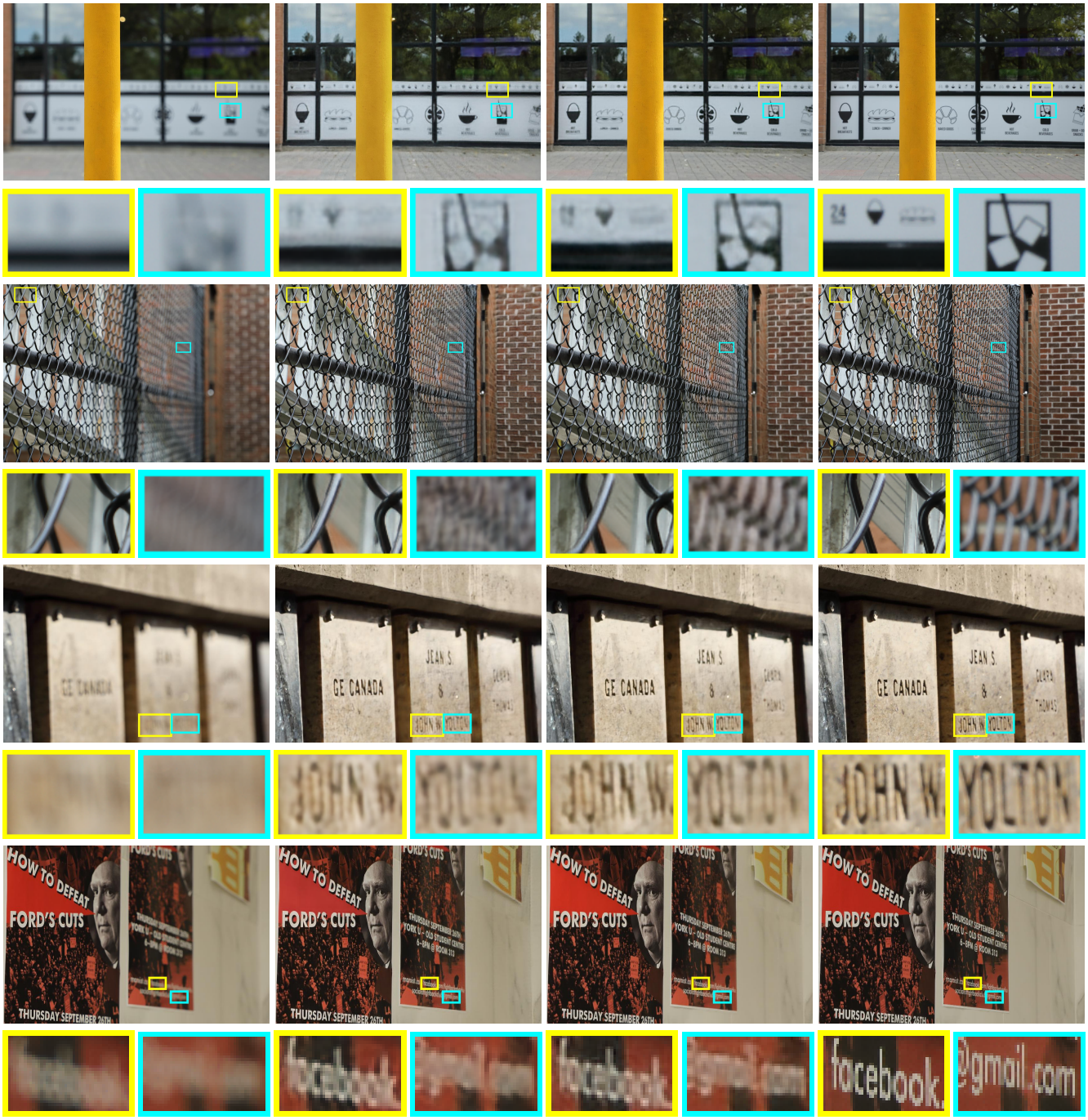
(a) input (b) DPDNet [1] (c) ours

Figure 9. Additional qualitative comparisons with DPDNet [1] on the defocused images in the CUHK blur detection dataset [5].



(a) input (b) DPDNet [1] (c) ours

Figure 10. Additional qualitative comparisons with DPDNet [1] on the defocused images in the CUHK blur detection dataset [5].



(a) input (b) DPDNet (dual) [1] (c) ours (dual) (d) GT

Figure 11. Qualitative comparisons of defocus deblurring models using dual-pixel input on the test set of the DPDD dataset [1].