Efficient Frequency Domain-based Transformers for High-Quality Image Deblurring - Supplemental Material -

Lingshun Kong[1], Jiangxin Dong[1*], Jianjun Ge[2], Mingqiang Li[2], and Jinshan Pan[1*]
[1]School of Computer Science and Engineering, Nanjing University of Science and Technology
[2]Information Science Academy, China Electronics Technology Group Corporation

Overview

In this document, we first provide further analysis on the proposed method in Section 1. Then, we show that the proposed method can be applied to other low-level vision tasks in Section 2. Finally, we show more visual comparisons of the proposed method and state-of-the-art ones in Section 3.

1. Further Analysis on the Proposed Method

As stated in Section 5 of the main manuscript, we have analyzed the effectiveness of the proposed frequency domain-based self-attention solver (FSAS) and the discriminative frequency domain-based feed-forward network on image deblurring. We have shown that using these proposed components achieves favorable performance against state-of-the-art methods on benchmarks while requiring lower computational costs and GPU memory.

Effect of the asymmetric encoder-decoder network. In Table 6 of the main manuscript, we have shown that using the proposed FSAS in both the encoder module and decoder module affects the final deblurred performance because the shallow features extracted by the encoder module usually contain blur effects that affect the estimations of the FSAS. In this document, we further compare the proposed method with the method only using the FSAS in the encoder module. Table 1 shows that only using the FSAS in the encoder module does not generate better results compared to the proposed method (see comparisons of "FSAS in enc" and "FSAS in dec (Ours)"). Figure 1 further shows that using the FSAS in the encoder module does not remove the blur effect well.

Table 1. Quantitative evaluations of the asymmetric encoder-decoder network on the GoPro dataset. Using the FSAS only in the encoder module or both in the encoder and decoder modules does not have better results compared to the proposed method using the FSAS only in the decoder module.

Methods	FSAS in enc	FSAS in enc&dec	FSAS in dec (Ours)
PSNRs	30.27	33.56	33.73
SSIMs	0.9337	0.9653	0.9663

Effect of the size of the quantization matrix. In our implementation, we use the patch size of 8×8 for the quantization matrix. We further examine the effect of the patch size on image deblurring, we Table 2 shows the effect of the quantization matrix size on the GoPro dataset. Given the performance and model complexity, we use 8×8 in the paper.

Memory analysis of the FSAS. In Table 4 of the main manuscript, we have shown that the proposed FSAS needs a small GPU memory. In this document, we explain why the maximum GPU memory of the fast Fourier transform (FFT) decreases with the window size in Table 4 of the main manuscript. We test the maximum GPU memory of the FFT separately which is

*Corresponding author: Jiangxin Dong and Jinshan Pan.

(a) Blurred image (b) FSAS in enc (c) FSAS in dec

Figure 1. Effect of the asymmetric encoder-decoder network. Using the proposed FSAS in the encoder module does not remove the blur effect well as shown in (b). In contrast, using the FSAS in the decoder module generates clearer images as shown in (c).

computed by the "torch.cuda.max_memory_allocated()" function. The size of the test tensor X is $[1, 64, 1024, 1024]$. We

Table 2. Effect of the size of the quantization matrix.

Size of the the quantization matrix	4×4	8×8	16×16
SSIMs	33.64	33.73	33.65
SSIMs	0.9656	0.9663	0.9657
Parameters	15.5M	16.6M	20.8M

Table 3. Memory of the FFT with different window sizes.

Window size	4×4	8×8	16×16	32×32	64×64	128×128	256×256	512×512	1024×1024
GPU memory	1024M	896M	832M	800M	784M	776M	772M	772M	769M

rearrange X into X_r with the size of $\left[1, 64, \frac{1024}{\text{windows\_size}}, \frac{1024}{\text{windows\_size}}, \text{window\_size}, \text{window\_size}\right]$. Table 3 shows the maximum GPU memory when applying the FFT to X_r.

We note that the maximum GPU memory of the FFT decreases with the window_size. This is mainly because the management of GPU memory in PyTorch is a hierarchical structure. It means that even if we allocate 4KB memory, the program will take up 2MB memory, which means Xr with a smaller window size needs to request memory more frequently. Thus when the window size is smaller, the maximum GPU memory of the fast Fourier transform (FFT) is larger.

2. Other Applications

We further show that the proposed method can be applied to other related low-level vision tasks. Following the protocols of [14], we train and evaluate our method on the image deraining task. Table 4 shows that our method works well on image deraining.

Table 4. Evaluations of the proposed method on image deraining. The results are obtained from the Test100 dataset [16]

Methods	MPRNet [15]	KiT [5]	Restormer [14]	MAXIM [12]	Ours
PSNRs	30.27	30.26	32.00	31.17	31.40
SSIMs	0.897	0.904	0.923	0.922	0.919

3. More Experimental Results

In this section, we provide more visual comparisons of the proposed method and state-of-the-art ones on benchmarks.

References

[1] Liangyu Chen, Xiaojie Chu, Xiangyu Zhang, and Jian Sun. Simple baselines for image restoration. In *ECCV*, 2022. 5, 6, 7

[2] Sung-Jin Cho, Seo-Won Ji, Jun-Pyo Hong, Seung-Won Jung, and Sung-Jea Ko. Rethinking coarse-to-fine approach in single image deblurring. In *ICCV*, 2021. 5, 6, 7, 8, 9, 10, 11, 12, 13

[3] Xiaojie Chu, Liangyu Chen, , Chengpeng Chen, and Xin Lu. Improving image restoration by revisiting global information aggregation. In *ECCV*, 2022. 5, 6, 7, 8, 9, 10

[4] Orest Kupyn, Volodymyr Budzan, Mykola Mykhailych, Dmytro Mishkin, and Jiri Matas. Deblurgan: Blind motion deblurring using conditional adversarial networks. In *CVPR*, 2018. 11, 12, 13

[5] Hunsang Lee, Hyesong Choi, Kwanghoon Sohn, and Dongbo Min. KNN local attention for image restoration. In *CVPR*, 2022. 3

[6] Xintian Mao, Yiming Liu, Wei Shen, Qingli Li, and Yan Wang. Deep residual fourier transformation for single image deblurring. *arXiv preprint arXiv:2111.11745*, 2021. 11, 12, 13

[7] Seungjun Nah, Tae Hyun Kim, and Kyoung Mu Lee. Deep multi-scale convolutional neural network for dynamic scene deblurring. In *CVPR*, 2016. 5, 6, 7

[8] Jaesung Rim, Haeyun Lee, Jucheol Won, and Sunghyun Cho. Real-world blur dataset for learning and benchmarking deblurring algorithms. In *ECCV*, 2020. 11, 12, 13

[9] Ziyi Shen, Wenguan Wang, Xiankai Lu, Jianbing Shen, Haibin Ling, Tingfa Xu, and Ling Shao. Human-aware motion deblurring. In *ICCV*, 2019. 8, 9, 10

[10] Xin Tao, Hongyun Gao, Xiaoyong Shen, Jue Wang, and Jiaya Jia. Scale-recurrent network for deep image deblurring. In *CVPR*, 2018. 11, 12, 13

[11] Fu-Jen Tsai, Yan-Tsung Peng, Yen-Yu Lin, Chung-Chi Tsai, and Chia-Wen Lin. Stripformer: Strip transformer for fast image deblurring. In *ECCV*, 2022. 5, 6, 7, 8, 9, 10, 11, 12, 13

[12] Zhengzhong Tu, Hossein Talebi, Han Zhang, Feng Yang, Peyman Milanfar, Alan Bovik, and Yinxiao Li. MAXIM: Multi-axis mlp for image processing. In *CVPR*, 2022. 3

[13] Zhendong Wang, Xiaodong Cun, Jianmin Bao, Wengang Zhou, Jianzhuang Liu, and Houqiang Li. Uformer: A general u-shaped transformer for image restoration. In *CVPR*, 2022.

[14] Syed Waqas Zamir, Aditya Arora, Salman Khan, Munawar Hayat, Fahad Shahbaz Khan, and Ming-Hsuan Yang. Restormer: Efficient transformer for high-resolution image restoration. In *CVPR*, 2022. 3, 5, 6, 7, 8, 9, 10

[15] Syed Waqas Zamir, Aditya Arora, Salman Khan, Munawar Hayat, Fahad Shahbaz Khan, Ming-Hsuan Yang, and Ling Shao. Multi-stage progressive image restoration. In *CVPR*, 2021. 3, 8, 9, 10

[16] He Zhang, Vishwanath Sindagi, and Vishal M. Patel. Image de-raining using a conditional generative adversarial network. *IEEE TCSVT*, 2020. 3

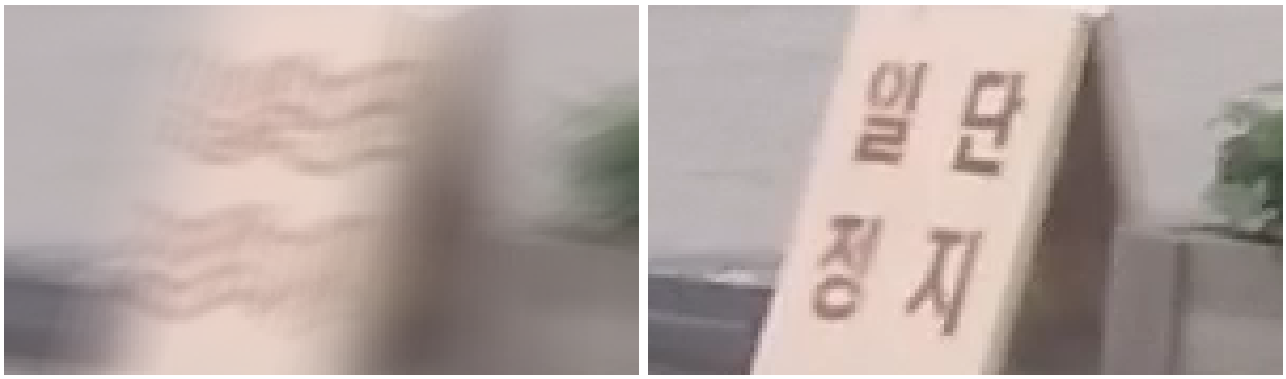
(a) Blurred image (b) GT

(c) MIMO-Unet+ [2] (d) Restormer [14]

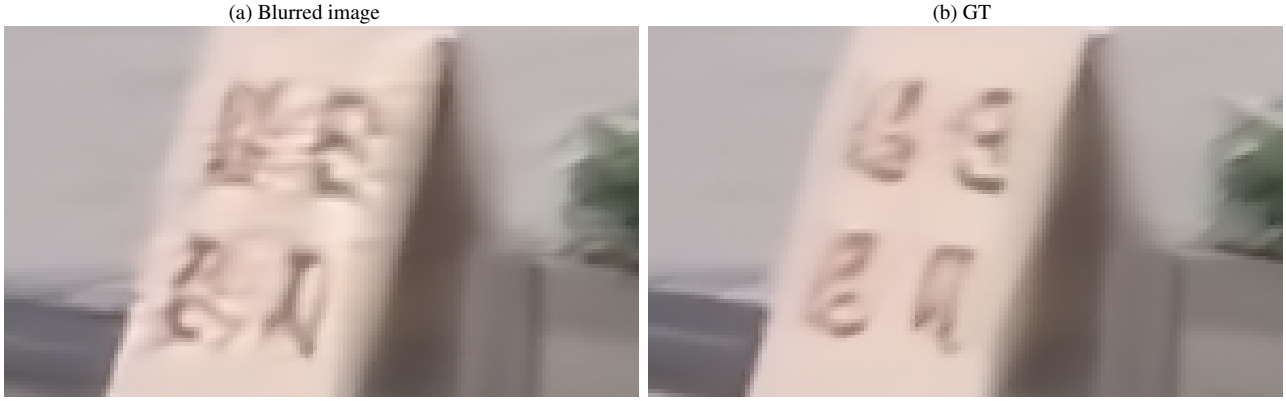
(e) Stripformer [11] (f) Restormer-local [3]

(g) NAFNet [1] (h) Ours

Figure 2. Deblurred results on the GoPro dataset [7]. The deblurred results in (c)-(g) still contain significant blur effects. The proposed method generates a clearer image, where the windows of the car are much clearer.

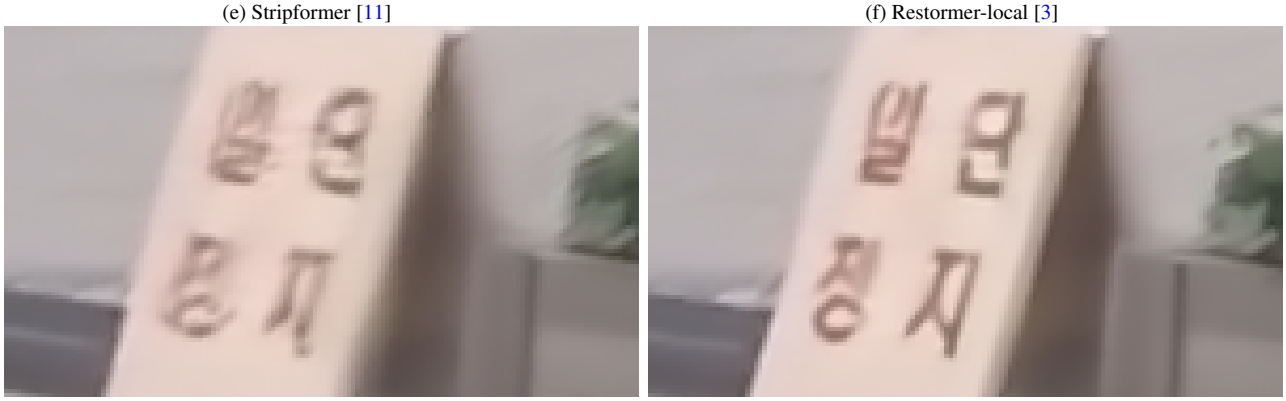


(a) Blurred image (b) GT



(c) MIMO-Unet+ [2] (d) Restormer [14]

(e) Stripformer [11] (f) Restormer-local [3]



(g) NAFNet [1] (h) Ours

Figure 3. Deblurred results on the GoPro dataset [7]. State-of-the-art methods [1–3, 11, 14] do not restore the characters well. In contrast, our method generates a clearer image with recognizable characters.

(a) Blurred image (b) GT

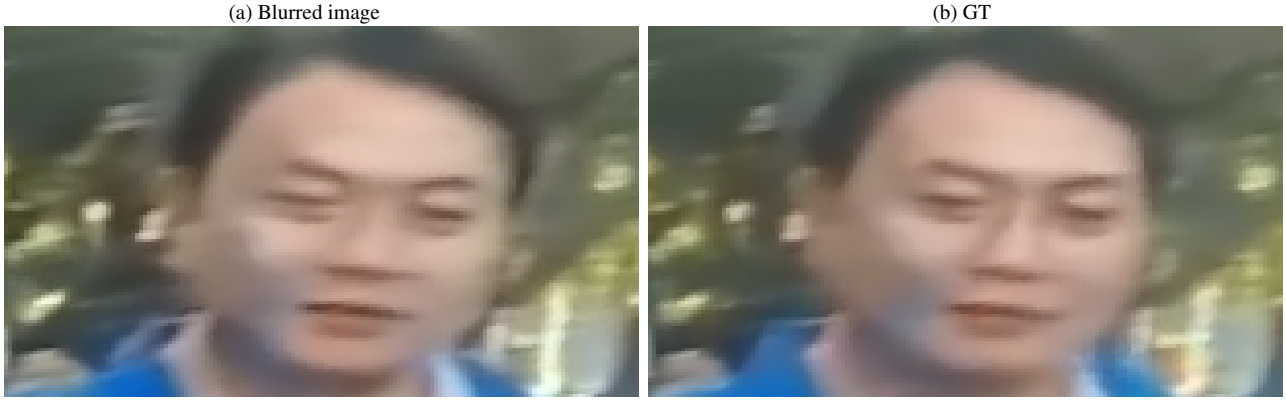
(c) MIMO-Unet+ [2] (d) Restormer [14]

(e) Stripformer [11] (f) Restormer-local [3]

(g) NAFNet [1] (h) Ours

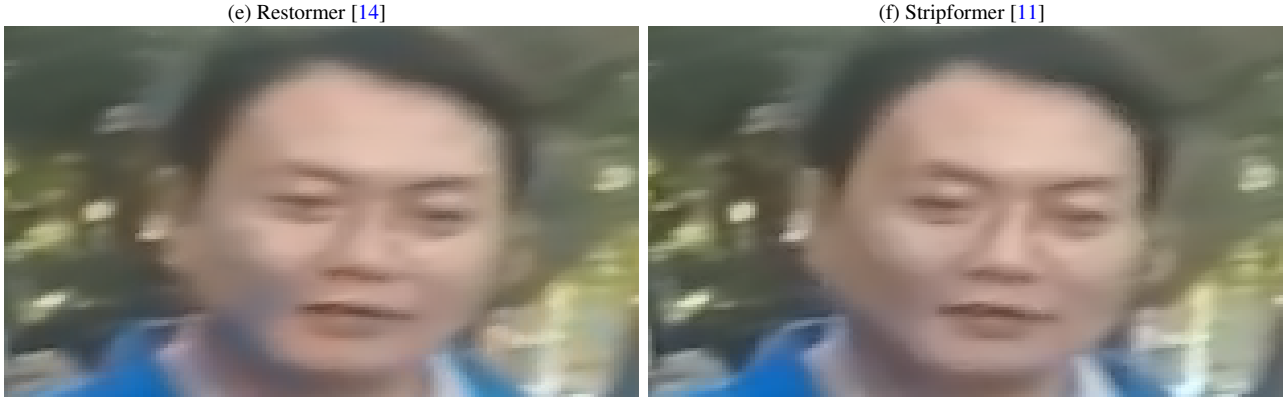
Figure 4. Deblurred results on the GoPro dataset [7]. State-of-the-art methods [1–3, 11, 14] do not restore the numbers well. In contrast, our method generates a clearer image, where the numbers are much clearer.

(a) Blurred image (b) GT



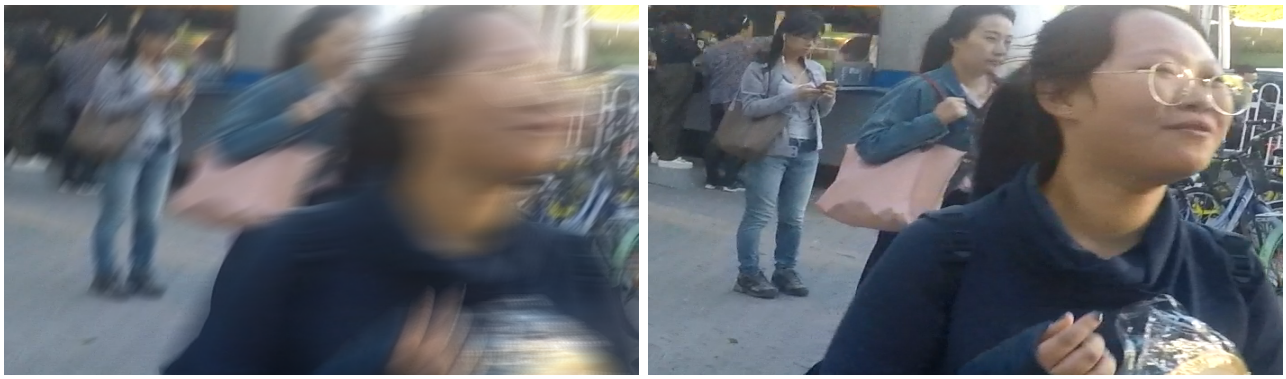
(c) MIMO-Unet+ [2] (d) MPRNet [15]

(e) Restormer [14] (f) Stripformer [11]

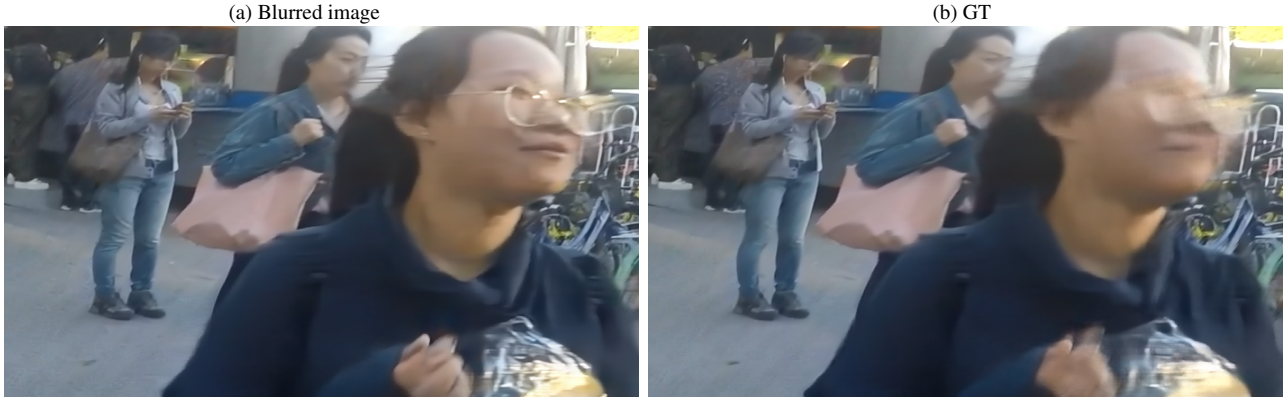


(g) Restormer-local [3] (h) Ours

Figure 5. Deblurred results on the HIDE dataset [9]. The deblurred results in (c)-(g) still contain artifacts in the face. The proposed method generates a clearer face image.

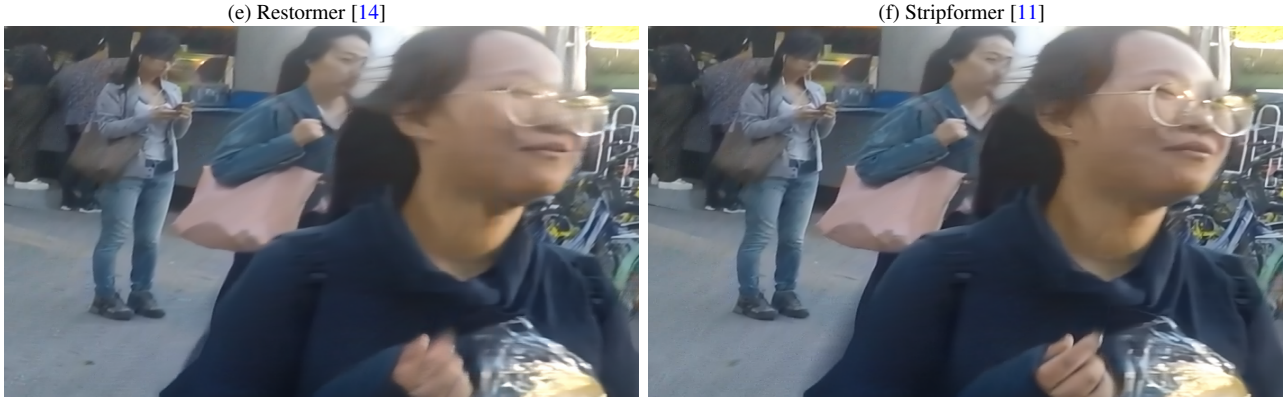


(a) Blurred image (b) GT



(c) MIMO-Unet+ [2] (d) MPRNet [15]

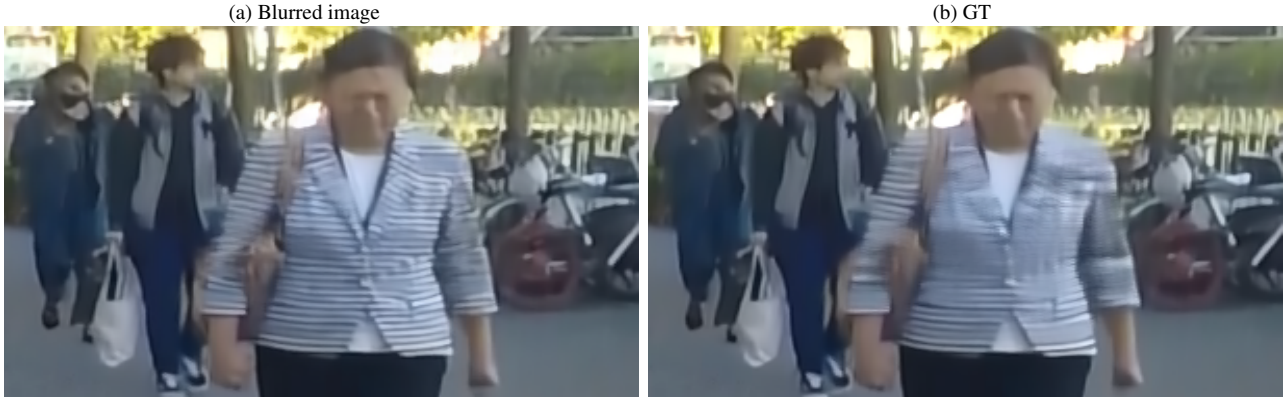
(e) Restormer [14] (f) Stripformer [11]



(g) Restormer-local [3] (h) Ours

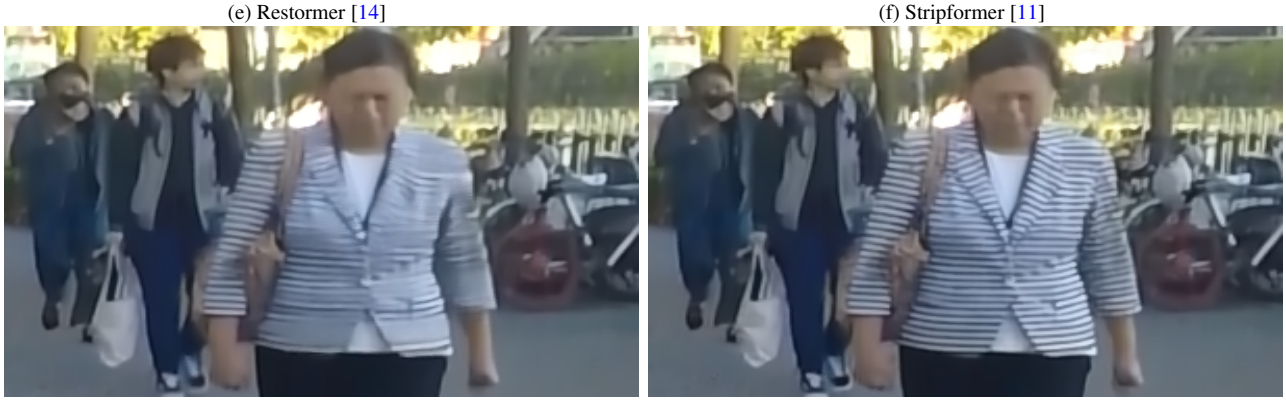
Figure 6. Deblurred results on the HIDE dataset [9]. The deblurred results in (c)-(g) still contain significant blur effects. The proposed method generates a clearer image. For example, the fingers and face are much clearer.

(a) Blurred image (b) GT



(c) MIMO-Unet+ [2] (d) MPRNet [15]

(e) Restormer [14] (f) Stripformer [11]



(g) Restormer-local [3] (h) Ours

Figure 7. Deblurred results on the HIDE dataset [9]. The deblurred results in (c)-(g) still contain significant blur effects. The proposed method generates a clearer image. For example, the the textures of clothes are much clearer.

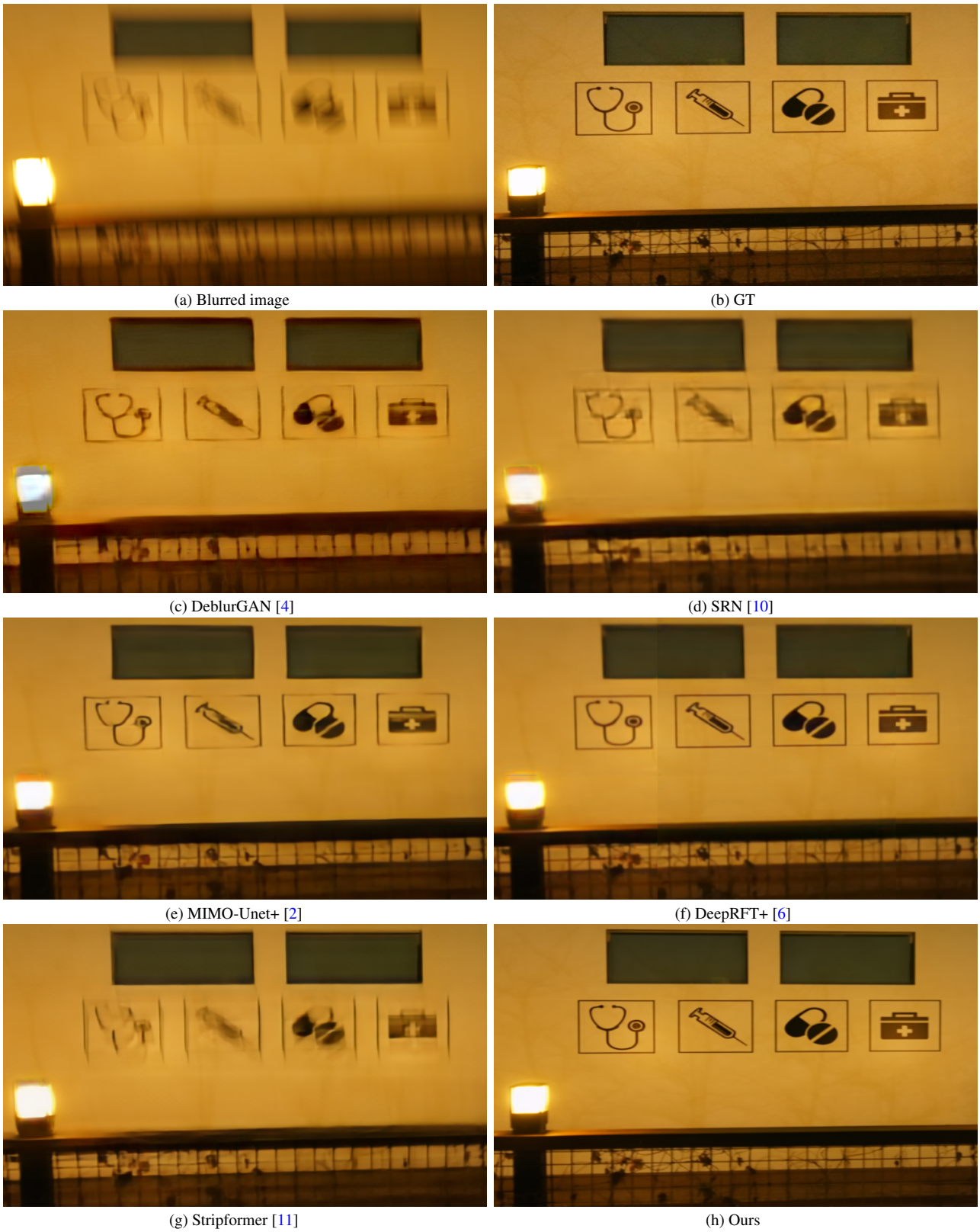
(a) Blurred image (b) GT

(c) DeblurGAN [4] (d) SRN [10]

(e) MIMO-Unet+ [2] (f) DeepRFT+ [6]

(g) Stripformer [11] (h) Ours

Figure 8. Deblurred results on the RealBlur dataset [8]. State-of-the-art methods [2, 4, 6, 10, 11] do not restore the characters well. In contrast, the proposed method generates a better image with clearer characters.



(a) Blurred image (b) GT

(c) DeblurGAN [4] (d) SRN [10]

(e) MIMO-Unet+ [2] (f) DeepRFT+ [6]

(g) Stripformer [11] (h) Ours

Figure 9. Deblurred results on the RealBlur dataset [8]. The deblurred results in (c)-(g) still contain significant blur effects. The proposed method generates a clearer image. For example, the icons and light are much clearer.

(a) Blurred image (b) GT

(c) DeblurGAN [4] (d) SRN [10]

(e) MIMO-Unet+ [2] (f) DeepRFT+ [6]

(g) Stripformer [11] (h) Ours

Figure 10. Deblurred results on the Realblur dataset [8]. The deblurred results in (c)-(g) still contain significant blur effects. The proposed method generates a clearer image. For example, the iron wire and the ladder are much clearer.