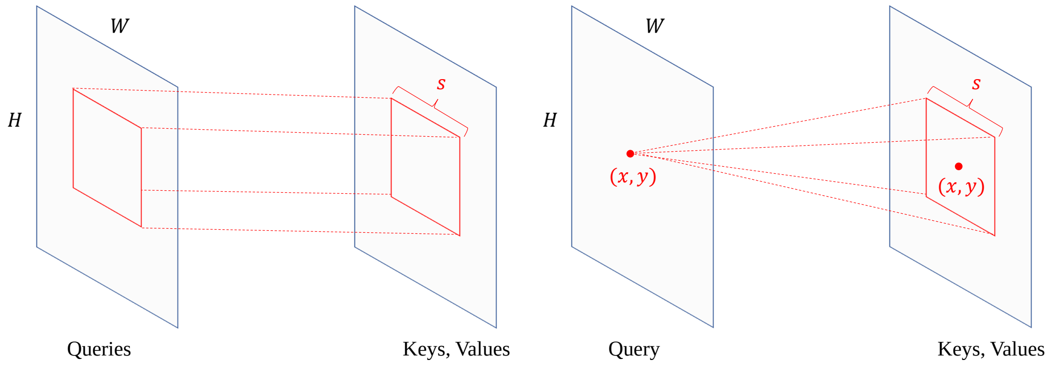
Supplemental Material for Stochastic Window Transformer for Image Restoration

Jie Xiao, Xueyang Fu[*], Feng Wu, Zheng-Jun Zha
University of Science and Technology of China, Hefei, China
ustchbxj@mail.ustc.edu.cn, {xyfu,fengwu,zhazj}@ustc.edu.cn

1 More Details

Small patch with global attention leads to the broken translation invariance and loss of locality. Due to the quadratic complexity with respect to the input resolution which is usually high for image restoration tasks, global attention applies only to the small patch in practice (e.g., 48×48 for IPT [2]). Under this setting, the broken translation invariance derives from two aspects. First, the absolute position encoding makes each token unique, which destroys the translation invariance [5, 4]. Second, the process of dividing small patches also incurs the broken translation invariance, which is similar to the aforementioned case of the fixed window partition. In a similar way, the dividing process also leads to the tremendous loss of locality.

Complexity of the sliding window strategy. For the sliding window strategy, every query corresponds to the distinct set of values and keys (Fig. 1(b)). This distinguishes from the fixed window strategy, in which all queries of the local window have the same set of values and keys (Fig. 1(a)). A direct consequence of the distinct context for every query is the huge memory overhead. Specifically, suppose the resolution of feature is (H, W) and the size of local window is $s \times s$, the memory footprint of K and V tensor for the sliding window strategy will be s^2 multiple of that for the fixed window strategy (compare Algorithm 1 with Algorithm 2). Since the resolution is often high for image restoration tasks, the huge memory usage often leads to the Out of Memory (OOM) problem in practice. Furthermore, as pointed out in [13, 9], although computational complexity is $\Theta(HW)$, the sliding window strategy is still slower in wall-clock time due to the lack of optimized kernels on various accelerators.



(a) The fixed window strategy (b) The sliding window strategy

Figure 1: The illustration of context of the shifted window strategy and sliding window strategy.

[*] Corresponding author.

Algorithm 1 Pytorch Implementation of the fixed window based attention

```
import torch.nn.functional as F
def FixedWindowAttention(x, win_size):
        C = x.shape[-1]
        x = F.unfold(x, kernel_size=win_size,stride=win_size)
        q, k, v = to_qkv(x)
        q = q * (C ** -0.5)
        attn = (q @ k.transpose(-2, -1))
        attn = softmax(attn + relative_position_bias)
        out = attn @ v
        return out
```

Algorithm 2 Pytorch Implementation of the sliding window based attention

```
import torch.nn.functional as F
def SlidingWindowAttention(x, win_size):
        C = x.shape[-1]
        q, k, v = to_qkv(x)
        k, v = pad(k), pad(v) # pad for keeping shape
        k = F.unfold(k, kernel_size=win_size, stride=1)
        #extra memory cost(win_size^2 X)
        v = F.unfold(v, kernel_size=win_size, stride=1)
        q = q * (C ** -0.5)
        attn = (q @ k.transpose(-2, -1))
        attn = softmax(attn + relative_position_bias)
        out = attn @ v
        return out
```

Taking expectation boosts performance. For simplicity, we denote $\{(\xi_h^l, \xi_w^l)\}_{l=0}^{N-1}$ collectively by ξ. We utilize the square of L^2 norm as the criterion to evaluate the fitted network. Therefore, given the degraded image x, the expected loss of the fitted $F(x, \xi)$ is given by

$$\begin{aligned}
&\mathop{\mathbb{E}}_{\xi}[||F(x,\xi) - I(x)||_2^2] && (1)\\
=&\mathop{\mathbb{E}}_{\xi}[||F(x,\xi) - \mathbb{E}_\xi[F(x,\xi)] + \mathbb{E}_\xi[F(x,\xi)] - I(x)||_2^2] && (2)\\
=&\mathbb{E}_\xi[||F(x,\xi) - \mathbb{E}_\xi[F(x,\xi)||_2^2] + ||\,\mathbb{E}_\xi[F(x,\xi)] - I(x)||_2^2 && (3)\\
\geq&||\,\mathbb{E}_\xi[F(x,\xi)] - I(x)||_2^2. && (4)
\end{aligned}$$

$I(x)$ is the ground-truth image of x. Derivation from (2) to (3) follows that

$$\mathbb{E}_\xi[< F(x,\xi) - \mathbb{E}_\xi[F(x,\xi)], \mathbb{E}_\xi[F(x,\xi)] - I(x) >] = 0. \quad (5)$$

$< \cdot, \cdot >$ is the inner product. Hence, we can readily draw the conclusion that taking expectation of the introduced stochastic shift, which corresponds to $\mathbb{E}_\xi[F(x,\xi)]$, helps to boost performance.

2 Experimental Setting

Image deraining. We train Stoformers using two Nvidia 3090 GPUs with batch size 8 on 256 × 256 image pairs. The training process lasts for 10 epochs. Following previous works [15, 18], We evaluate PSNR [7] and SSIM [17] based on the luminance channel, i.e., Y channel of YCbCr space.

Image denosing. Following [20], we construct a large dataset comprising 400 BSD images [3], $4,744$ Waterloo Exploration Database images [10], 900 DIV2K images [1] and $2,750$ Flick2K images [8] for training. To tackle with a range of noise levels, the training images are corrupted by Gaussian noise with σ randomly chose from $[0, 50]$. The training patches are cropped from the total training set with size 128 × 128. We train Stoformers using two Nvidia 3090 GPUs for total 120 epoches with batch size 16 and PSNR is evaluated on the full-size test images.

Image deblurring. Stoformers are trained on GoPro dataset [12][2] and directly applied to GoPro [12] and HIDE [14]. We crop 512 × 512 image patches with stride 256 from GoPro dataset and train

[2] `https://seungjunnah.github.io/Datasets/gopro`, CC BY 4.0 license.

Stoformers with 256×256 training pairs randomly cropped from 512×512 image patches. The total training epoch is 600 with batch size 8 on two Nvidia 3090 GPUs and we evaluate PSNR and SSIM on the full-size test images.

3 Visualization

3.1 Feature Map

Fig. 2 presents more visualization of feature maps from various depth of the stochastic window transformer and fixed window transformer. Feature maps from the fixed window transformer contain obvious blocking artifacts due to the lack of translation invariance. In contrast, the stochastic window transformer utilizes the stochastic window strategy to accomplish the translation invariance so that these artificial blocking artifacts can be removed significantly.

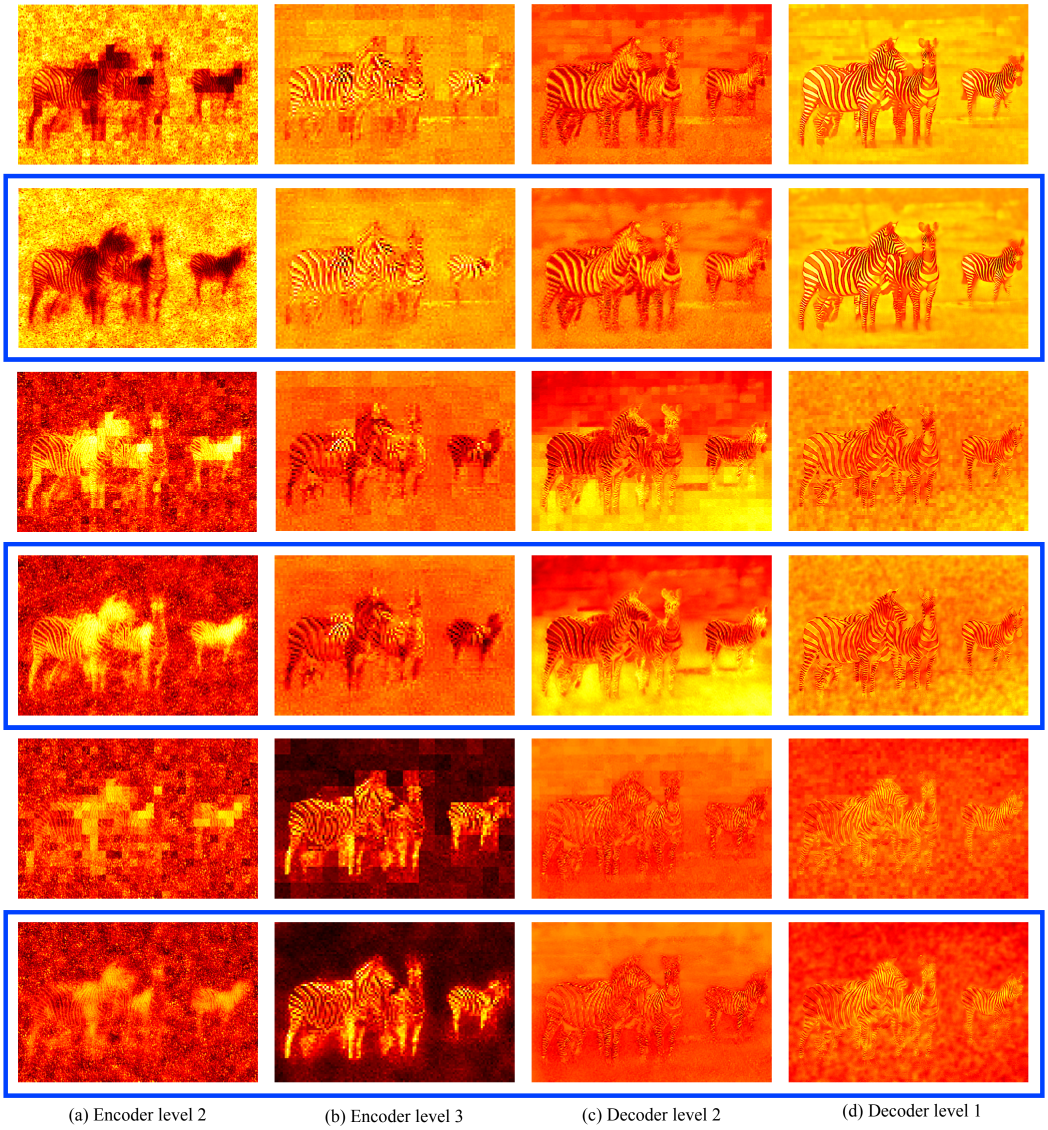
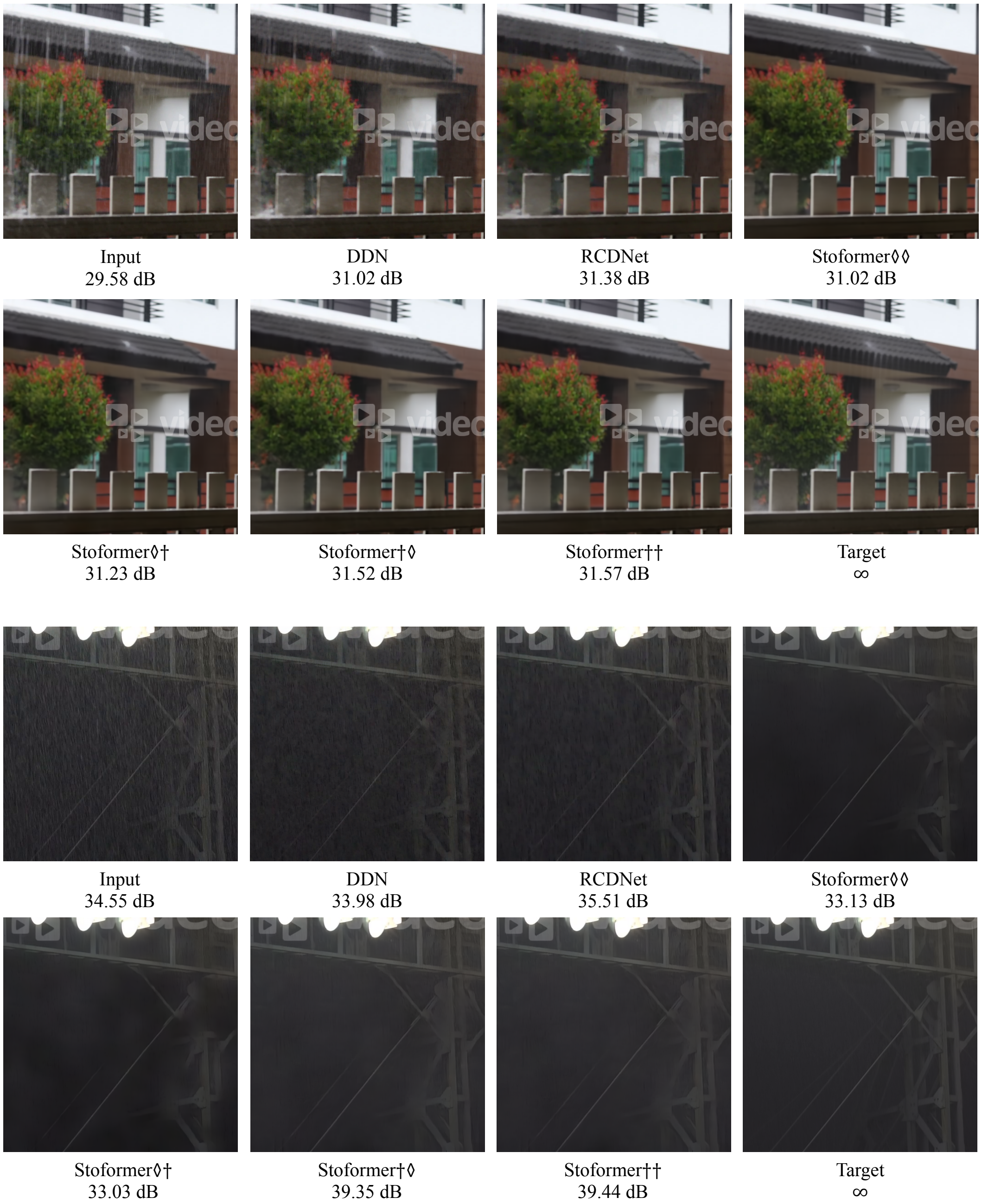

(a) Encoder level 2 (b) Encoder level 3 (c) Decoder level 2 (d) Decoder level 1

Figure 2: Feature maps from various depth of the denoising network. The feature maps in the blue box are from the stochastic window transformer while others are taken from the fixed window transformer. Please zoom in for better visualization.

3.2 Image Restoration Result

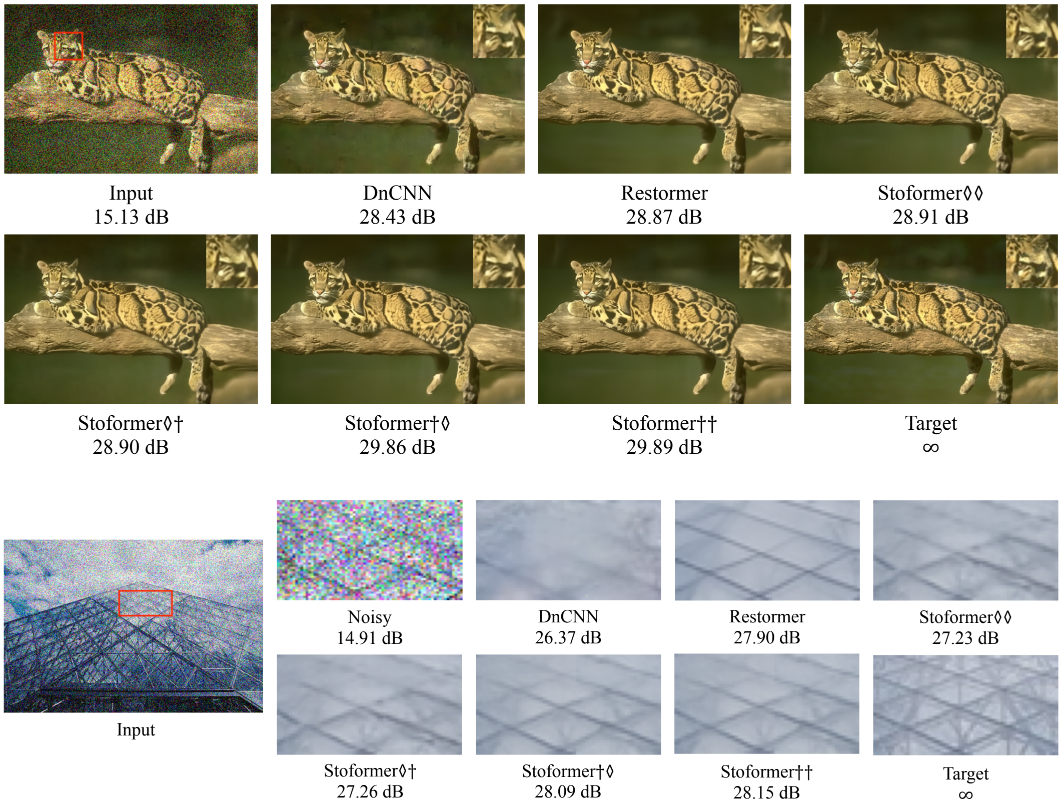
We also provide more visual results on image deraining (Fig. 3), image denoising (Figs. 4 and 5 for color images and Figs. 6 and 7 for grayscale images), and image deblurring (Figs. 8 and 9). In comparison with other state-of-the-art methods and Stoformer variants, Stoformer††, which is equipped with the stochastic window for training and layer expectation propagation for testing, can recover more image textures and further generate visually faithful results.



Input 29.58 dB
DDN 31.02 dB
RCDNet 31.38 dB
Stoformer◊◊ 31.02 dB
Stoformer◊† 31.23 dB
Stoformer†◊ 31.52 dB
Stoformer†† 31.57 dB
Target ∞

Input 34.55 dB
DDN 33.98 dB
RCDNet 35.51 dB
Stoformer◊◊ 33.13 dB
Stoformer◊† 33.03 dB
Stoformer†◊ 39.35 dB
Stoformer†† 39.44 dB
Target ∞

Figure 3: Visual comparison of image deraining on the SPA-Data [16].



Input
15.13 dB

DnCNN
28.43 dB

Restormer
28.87 dB

Stoformer◊◊
28.91 dB

Stoformer◊†
28.90 dB

Stoformer†◊
29.86 dB

Stoformer††
29.89 dB

Target
∞

Input

Noisy
14.91 dB

DnCNN
26.37 dB

Restormer
27.90 dB

Stoformer◊◊
27.23 dB

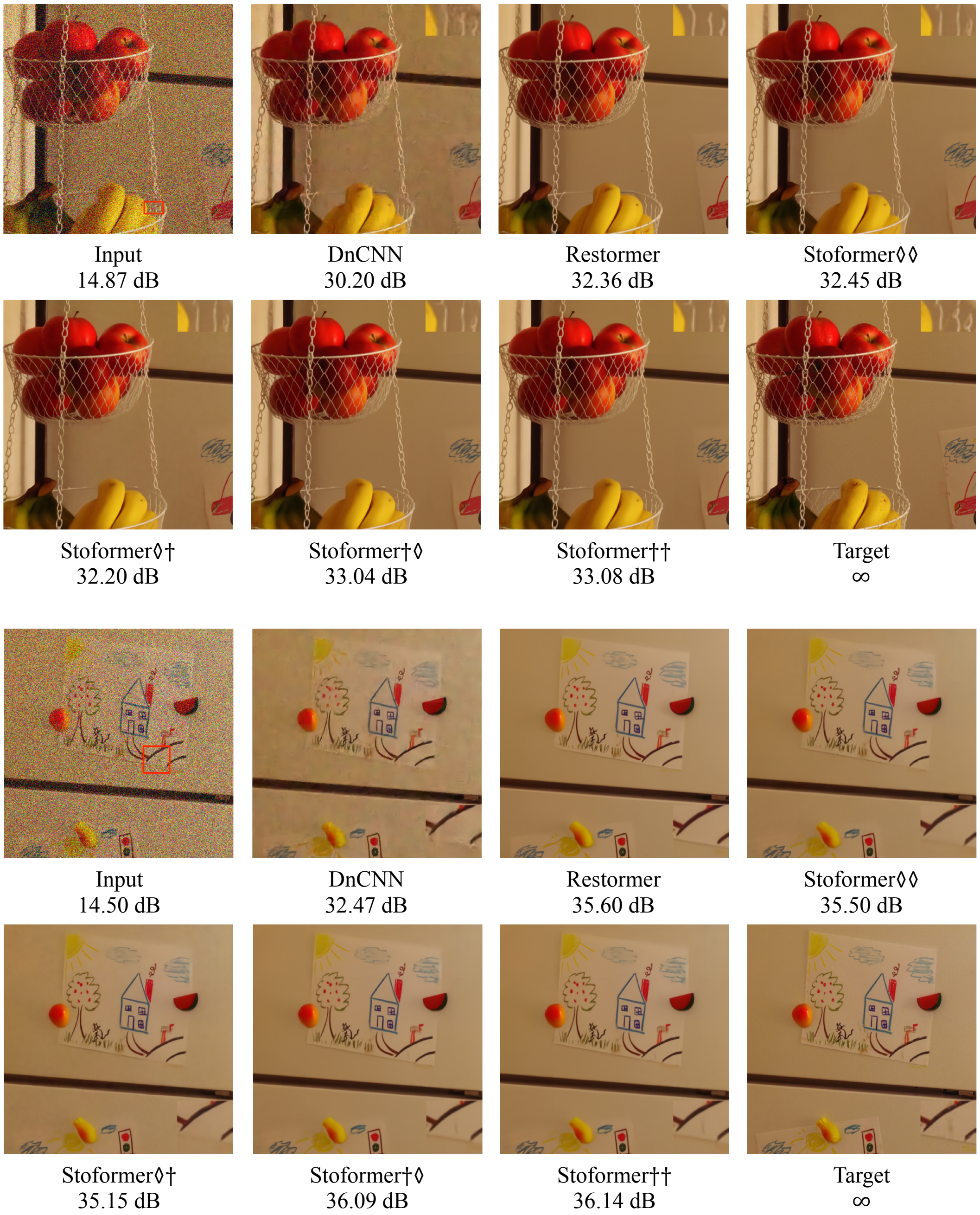
Stoformer◊†
27.26 dB

Stoformer†◊
28.09 dB

Stoformer††
28.15 dB

Target
∞

Figure 4: Visual comparison of Gaussian color denoising on the BSD68 [11].



Input
14.87 dB

DnCNN
30.20 dB

Restormer
32.36 dB

Stoformer◊◊
32.45 dB

Stoformer◊†
32.20 dB

Stoformer†◊
33.04 dB

Stoformer††
33.08 dB

Target
∞

Input
14.50 dB

DnCNN
32.47 dB

Restormer
35.60 dB

Stoformer◊◊
35.50 dB

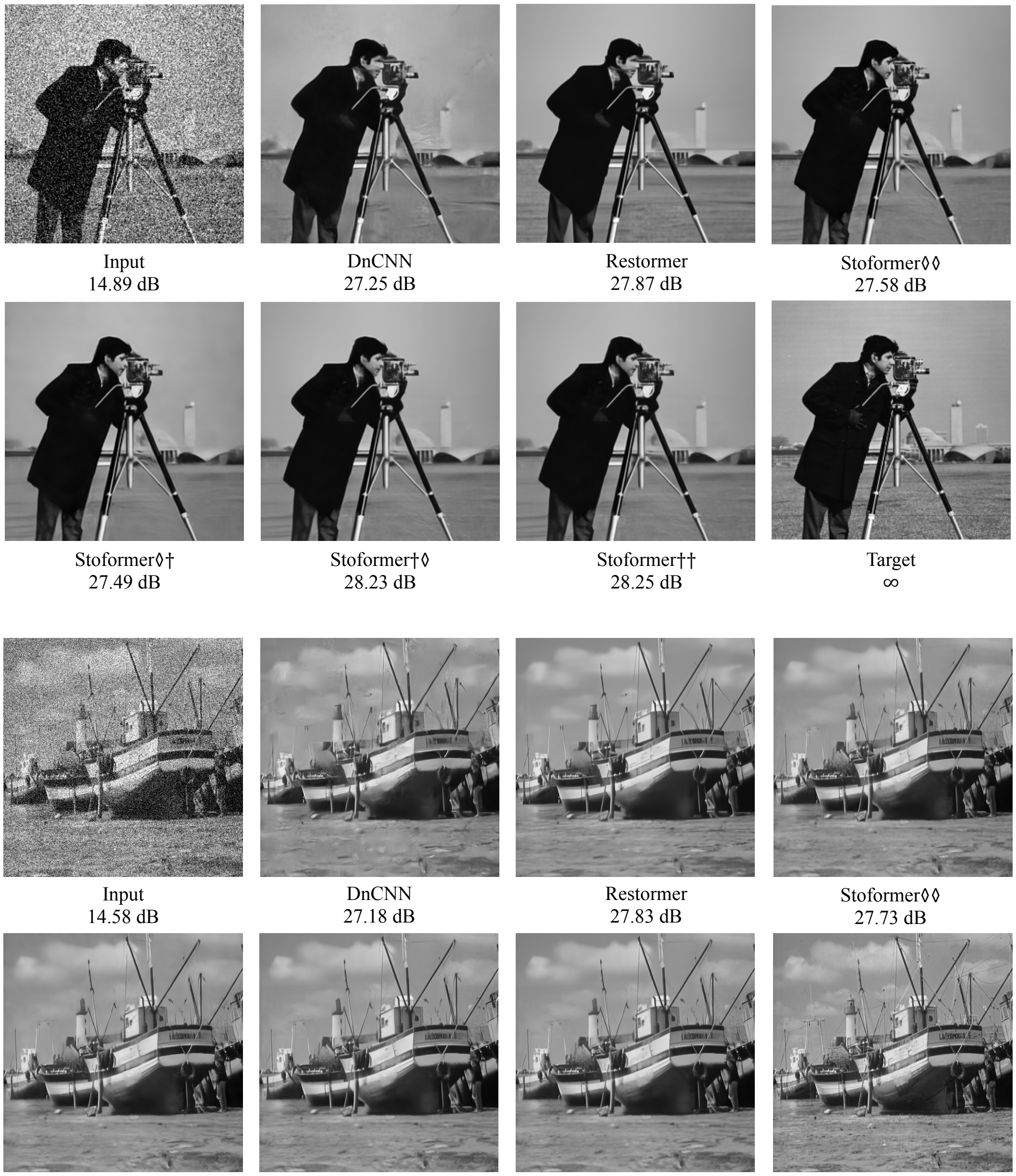
Stoformer◊†
35.15 dB

Stoformer†◊
36.09 dB

Stoformer††
36.14 dB

Target
∞

Figure 5: Visual comparison of Gaussian color image denoising on the McMaster68 [21].



Input
14.89 dB

DnCNN
27.25 dB

Restormer
27.87 dB

Stoformer◊◊
27.58 dB

Stoformer◊†
27.49 dB

Stoformer†◊
28.23 dB

Stoformer††
28.25 dB

Target
∞

Input
14.58 dB

DnCNN
27.18 dB

Restormer
27.83 dB

Stoformer◊◊
27.73 dB

Stoformer◊†
27.70 dB

Stoformer†◊
27.95 dB

Stoformer††
27.98 dB

Target
∞

Figure 6: Visual comparison of Gaussian grayscale image denosing on Set12 [19].

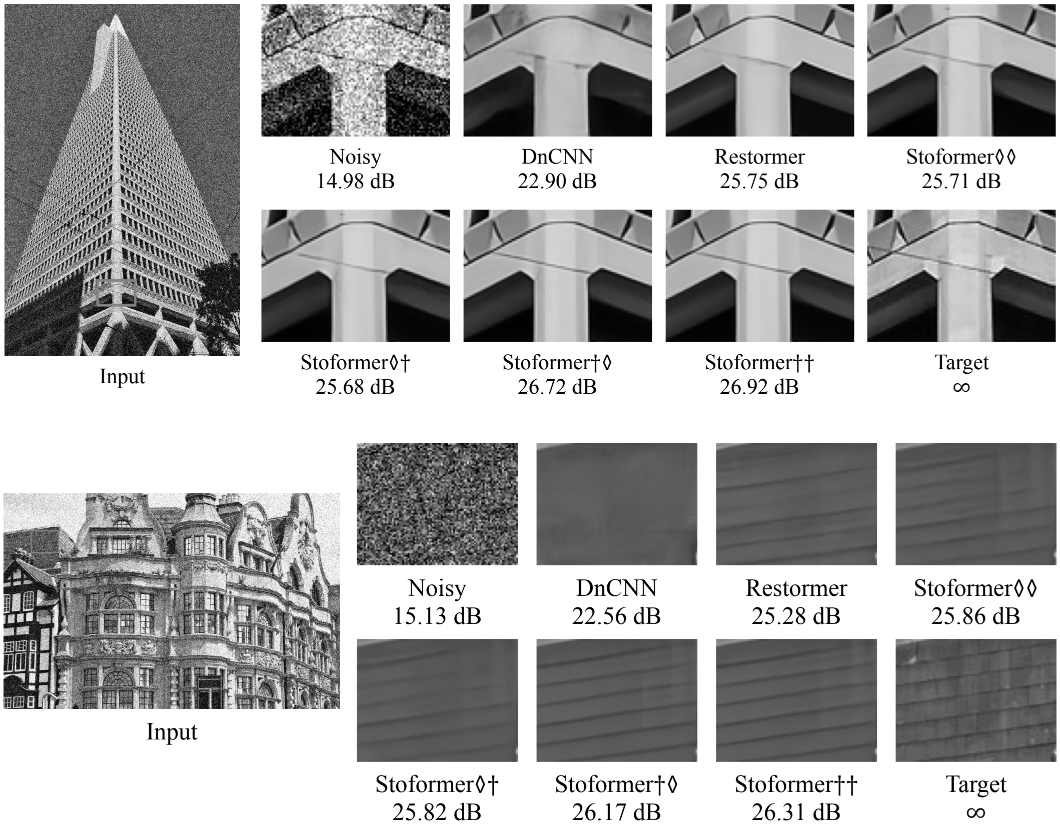


Figure 7: Visual comparison of Gaussian grayscale image denosing on Urban100 [6].

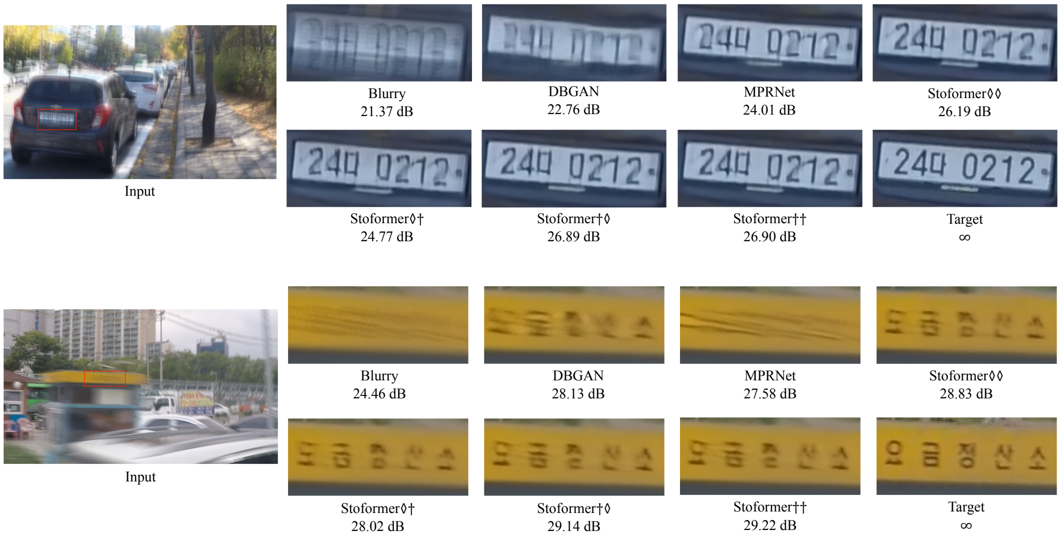
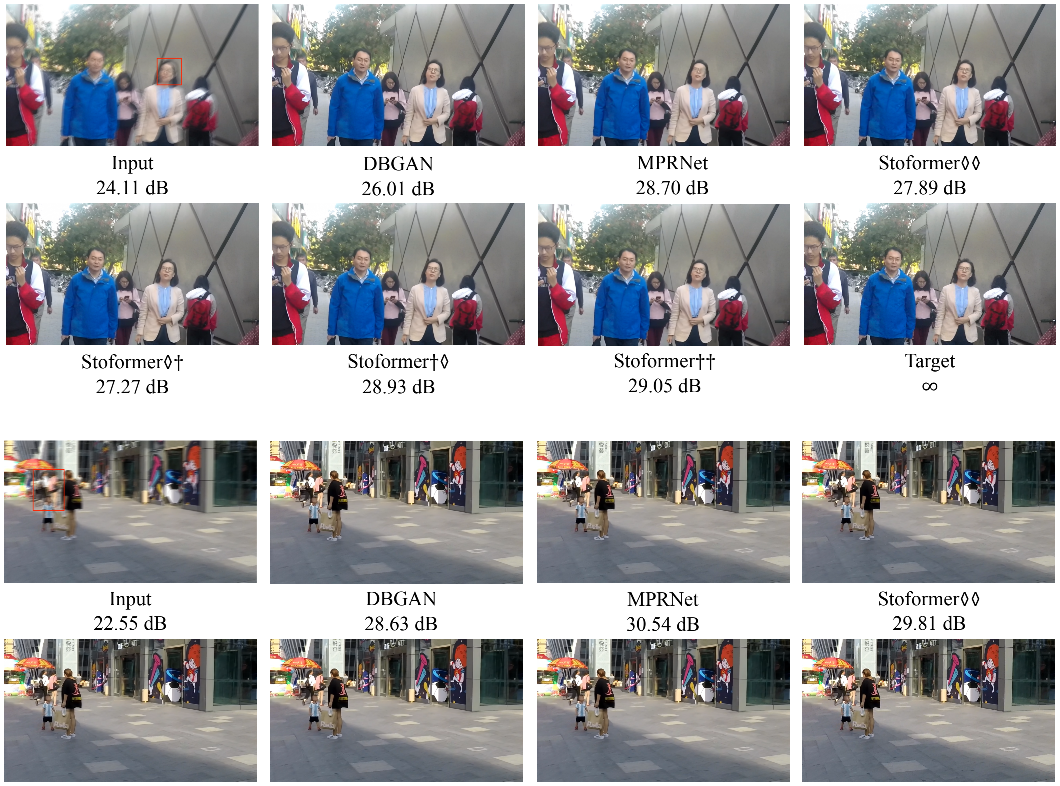


Figure 8: Visual comparison of image deblurring on the GoPro [12].



Input 24.11 dB | DBGAN 26.01 dB | MPRNet 28.70 dB | Stoformer◊◊ 27.89 dB

Stoformer◊† 27.27 dB | Stoformer†◊ 28.93 dB | Stoformer†† 29.05 dB | Target ∞

Input 22.55 dB | DBGAN 28.63 dB | MPRNet 30.54 dB | Stoformer◊◊ 29.81 dB

Stoformer◊† 28.75 dB | Stoformer†◊ 30.51 dB | Stoformer†† 30.56 dB | Target ∞

Figure 9: Visual comparison of image deblurring on the HIDE [14].

References

[1] E. Agustsson and R. Timofte. Ntire 2017 challenge on single image super-resolution: Dataset and study. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition Workshops*, 2017.

[2] H. Chen, Y. Wang, T. Guo, C. Xu, Y. Deng, Z. Liu, S. Ma, C. Xu, C. Xu, and W. Gao. Pre-trained image processing transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021.

[3] Y. Chen and T. Pock. Trainable nonlinear reaction diffusion: A flexible framework for fast and effective image restoration. *IEEE transactions on Pattern Analysis and Machine Intelligence*, 39(6):1256–1272, 2016.

[4] X. Chu, Z. Tian, B. Zhang, X. Wang, X. Wei, H. Xia, and C. Shen. Conditional positional encodings for vision transformers. *arXiv preprint arXiv:2102.10882*, 2021.

[5] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2021.

[6] J.-B. Huang, A. Singh, and N. Ahuja. Single image super-resolution from transformed self-exemplars. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, 2015.

[7] Q. Huynh-Thu and M. Ghanbari. Scope of validity of psnr in image/video quality assessment. *Electronics letters*, 44(13):800–801, 2008.

[8] B. Lim, S. Son, H. Kim, S. Nah, and K. Mu Lee. Enhanced deep residual networks for single image super-resolution. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition Workshops*, 2017.

[9] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, and B. Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021.

[10] K. Ma, Z. Duanmu, Q. Wu, Z. Wang, H. Yong, H. Li, and L. Zhang. Waterloo exploration database: New challenges for image quality assessment models. *IEEE Transactions on Image Processing*, 26(2):1004–1016, 2016.

[11] D. Martin, C. Fowlkes, D. Tal, and J. Malik. A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In *Proceedings Eighth IEEE International Conference on Computer Vision*, 2001.

[12] S. Nah, T. Hyun Kim, and K. Mu Lee. Deep multi-scale convolutional neural network for dynamic scene deblurring. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, 2017.

[13] P. Ramachandran, N. Parmar, A. Vaswani, I. Bello, A. Levskaya, and J. Shlens. Stand-alone self-attention in vision models. *Advances in Neural Information Processing Systems*, 2019.

[14] Z. Shen, W. Wang, X. Lu, J. Shen, H. Ling, T. Xu, and L. Shao. Human-aware motion deblurring. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019.

[15] H. Wang, Q. Xie, Q. Zhao, and D. Meng. A model-driven deep neural network for single image rain removal. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020.

[16] T. Wang, X. Yang, K. Xu, S. Chen, Q. Zhang, and R. W. Lau. Spatial attentive single-image deraining with a high quality real rain dataset. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019.

[17] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13(4):600–612, 2004.

[18] W. Yang, R. T. Tan, J. Feng, J. Liu, Z. Guo, and S. Yan. Deep joint rain detection and removal from a single image. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017.

[19] K. Zhang, W. Zuo, Y. Chen, D. Meng, and L. Zhang. Beyond a gaussian denoiser: Residual learning of deep cnn for image denoising. *IEEE Transactions on Image Processing*, 26(7): 3142–3155, 2017.

[20] K. Zhang, Y. Li, W. Zuo, L. Zhang, L. Van Gool, and R. Timofte. Plug-and-play image restoration with deep denoiser prior. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2021.

[21] L. Zhang, X. Wu, A. Buades, and X. Li. Color demosaicking by local directional interpolation and nonlocal adaptive thresholding. *Journal of Electronic imaging*, 20(2):023016, 2011.