Hybrid Spectral Denoising Transformer with Guided Attention Supplementary Material

Zeqiang Lai[1], Chenggang Yan[2], Ying Fu[1†]
[1]Beijing Institute of Technology [2] Hangzhou Dianzi University
{laizeqiang, fuying}@bit.edu.cn cgyan@hdu.edu.cn

A. More Details about GSSA

Computational Complexity of GSSA. Given an input $\mathbf{X} \in \mathbb{R}^{H\times W\times D\times C}$ where H, W denote height and width, D denotes the number of spectral bands, C denotes the features channels, the computational complexity of each step of GSSA is summarized in Tab. 1. Since the feature channels are typically larger than the number of spectral bands, the asymptotic computational complexity of GSSA is dominated by two linear transformations, *i.e.*, *Linear for V* and *Post linear*.

Step	Complexity
Linear for V	$(H \times W \times D) \times C^2$
Pooling for Q,K	$2 \times (H \times W) \times (D \times C)$
Compute attention matrix	$D \times D \times C$
Feature aggregation	$H \times W \times C \times D \times D$
Post linear	$(H \times W \times D) \times C^2$
Total	$O((H \times W \times D) \times C^2)$

Table 1: The computational complexity of GSSA. The overall complexity of GSSA is linear with respect to image size.

Fast Implementation. With the simplification of pixel-wise attention via global average pooling, our GSSA can be efficiently implemented with a depth-wise convolution by treating the shared attention map as a convolution filter and swapping the spectral and channel dimensions. The speed comparison is shown in Tab. 2, and it can be seen that the Conv-based implementation is approximately 20% faster than the naive `Matmul`-based one.

Implementation	Runtime (s)	PSNR
Matmul-based	0.60	41.82
Conv-based	0.47	41.82

Table 2: Speed of different implementations of GSSA. Our Conv-based implementation reduces the running time without harming the performance.

A.1. Comparison against other Attention.

Here, we provide a more detailed explanation regarding the differences between our GSSA and existing channel or spectral attention mechanisms. *We highlight that our GSSA is significantly different from previous attention mechanisms in a variety aspects*. Since GSSA performs attention along spectral rather than spatial dimensions, we here compare it with four previous attention mechanisms that apply along spectral or channel dimensions including:

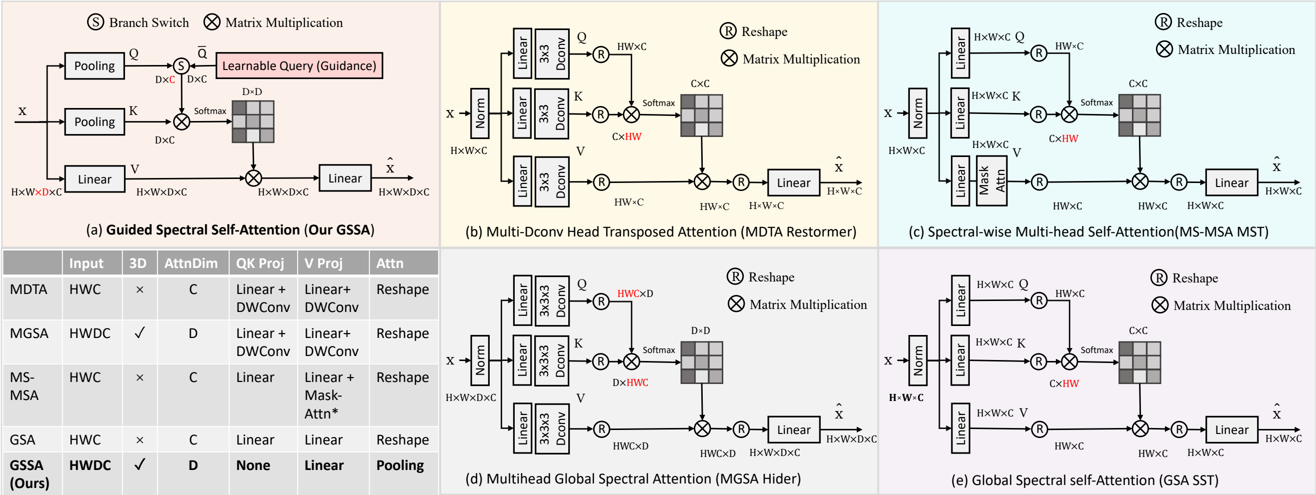
Attention	Method	Task
MDTA	Restormer [15]	Color image restoration
MS-MSA	MST [1]	Spectral Reconstruction
GSA	SST [9]	HSI denoising
MGSA	Hider [3]	HSI denoising

Table 3: The competing attention mechanisms.

Fig. 1 illustrates the structures of the aforementioned attention mechanisms. It is worth noting that all previous methods are essentially variants of MDTA proposed in Restormer, whereas our GSSA is fundamentally distinct from them. In the following, we will provide a detailed explanation of the main differences between the previous methods and our GSSA.

3D vs 2D Data Format. The first notable difference, which can be easily confused with previous work, is that *our GSSA performs attention on the spectral dimension*, i.e., the D dimension of a 5D data cube $x \in \mathbb{R}^{B\times C\times D\times H\times W}$. In contrast, previous works, such as MST, and SST, even though they refer to their attention mechanisms as spectral attention, essentially apply channel attention along the C dimension of a 4D data cube $x \in \mathbb{R}^{B\times C\times H\times W}$, which is the same as MDTA. Our 3D approach provides the flexibility to handle HSIs with different bands within a single model. Additionally, it achieves superior performance by preserving the structures of different bands, *i.e.*, each band possesses its own feature set, and their relationship remains unchanged across layers of the entire model.

† Corresponding Author.



(a) **Guided Spectral Self-Attention (Our GSSA)**

(b) Multi-Dconv Head Transposed Attention (MDTA Restormer)

(c) Spectral-wise Multi-head Self-Attention(MS-MSA MST)

	Input	3D	AttnDim	QK Proj	V Proj	Attn
MDTA	HWC	×	C	Linear + DWConv	Linear+ DWConv	Reshape
MGSA	HWDC	✓	D	Linear + DWConv	Linear+ DWConv	Reshape
MS-MSA	HWC	×	C	Linear	Linear + Mask-Attn*	Reshape
GSA	HWC	×	C	Linear	Linear	Reshape
GSSA (Ours)	**HWDC**	✓	**D**	**None**	**Linear**	**Pooling**

(d) Multihead Global Spectral Attention (MGSA Hider)

(e) Global Spectral self-Attention (GSA SST)

Figure 1: Comparison of different spectral/channel attention mechanisms. Our GSSA is significantly different from previous attention mechanisms. We could observe MDTA and MGSA are almost identical; MS-MSA and GSA are almost identical. Besides, MS-MSA and GSA are also basically simpler version of MDTA without depthwise convolution. Please refer to the text for detailed explanation.

QKV Projection. The second key difference pertains to the projections used for the query, key, and value. Conventional attention mechanisms typically employ three linear projections to project the input into query, key, and value. This approach is utilized in all of the compared methods, with the exception of our GSSA. Instead, *our GSSA applies linear projection solely for the value, which greatly simplifies the design*. By contrast, MDTA needs a extra 3x3 depthwise convolution after the linear projection. MGSA is identical to MDTA, except that it employs a 3D convolution. MS-MSA and GSA are the same and solely utilize linear projections, with the exception that MS-MSA employs an additional mask attention specifically designed for spectral reconstruction.

Pooling vs Reshape. The third difference is that our GSSA uses global average pooling to obtain feature maps for each band. This differs from previous methods that adopt a reshape approach. Our method is significantly more computationally efficient compared to previous approaches. Previous methods reshape the Q, K, and V tensors from a shape of $H \times W \times C$ into $HW \times C$, treating HW as the features for each channel. This leads to a time complexity of dot-product attention that is linear with respect to the image size, *i.e.*, $D \times D \times HWC$. In contrast, our GSSA approach only has a constant time complexity $D \times D \times C$, where D denotes the number of bands.

Learnable Query. The fourth notable difference is the introduction of the learnable query (LQ), which is motivated by the fixed patterns of pixel values across different bands. For example, the values of band 100 nm and 200 nm are correlated. Our LQ helps to identify these correlations and the alternative training strategy enables improvements without any extra cost on the number of parameters, inference time, and the flexibility to handle HSIs with different bands.

Model	#P(Conv)	#P(Total)	PSNR	SAM
Conv3D	0.43M	0.58M	41.62	0.052
Sep3D [4]	0.37M	0.53M	41.44	0.054
S3Conv-S	0.26M	0.42M	41.47	0.052
S3Conv-Seq	0.26M	0.42M	41.58	0.052
S3Conv	0.36M	0.52M	**41.82**	**0.049**

Table 4: Comparison of different S3Conv variants against 3D convolution and previous separable convolution. Our S3Conv achieves significant better performance with fewer parameters. Our methods are highlighted as gray . #P denotes the model parameters.

B. More Ablation Studies

To evaluate the effectiveness of the proposed components, we conduct a series of experiments to explore the different design choices for each part of our HSDT architecture. Specifically, we compare the proposed blocks, which include GSSA, S3Conv, and SM-FNN, by separately replacing them with existing blocks that share the same functionality, *e.g.*, replacing S3Conv with Conv3D. We use HSDT-M as the base model and evaluate the performance of the different blocks by replacing them one at a time. For blocks that cannot be incorporated into our 3D architectural design of HSDT, such as 2D spectral attention [9], we report the results obtained using their respective models.

Spatial-Spectral Separable Convolution. We evaluate several variants of our S3Conv. The most straightforward variant, S3Conv-S, sets the number of spatial convolutions to 1, while the S3Conv variant that we adopt uses 2. Another variant, S3Conv-Seq, applies spatial and spectral convolutions sequentially instead of in parallel. As shown in Table 4, both variants achieve comparable performance with roughly 60% of the parameters used by Conv3D. Our adopted version achieves a 0.2 dB PSNR gain with only 80% of the parameters used by Conv3D. Notably, our S3Conv approach significantly outperforms previous HSI separable convolution approaches [4], achieving over 0.4 dB PSNR improvement with even fewer parameters.

Guided Spectral Self-Attention. We compare the proposed GSSA approach with existing spectral fusion techniques, including QRU [14], GSA [9], MS-MSA [1], MDTA [15], and MGSA [3]. It is worth noting that although GSA and MS-MSA are named as spectral attention, they are essentially channel attentions derived from MDTA, as discussed earlier. Furthermore, GSA, MS-MSA, and MDTA are all 2D attention approaches that work with 4D data formats instead of the 5D data format used by HSDT. Therefore, we report the results of their models when compared with GSA, MS-MSA, and MDTA. For 3D spectral fusion techniques such as QRU and MGSA, we report the results of models that replace the GSSA of HSDT-M with them. Table 5 presents the results of different attention mechanisms. Our GSSA approach achieves the best results against the other approaches. Notably, our GSSA outperforms previous GSA and MGSA approaches (which are also designed for HSI denoising) by a large margin, demonstrating the effectiveness of our designs.

Model	Params	PSNR	SAM
QRU [14]	0.57M	41.31	0.064
GSA [9] & MS-MSA [1]	4.14M	41.41	0.052
MDTA [15]	26.2M	41.03	0.062
MGSA [3]	0.50M	39.74	0.102
GSSA	0.52M	**41.82**	**0.049**

Table 5: Results of our GSSA in comparison with other attention blocks. Our GSSA achieves a prominent improvement against QRU by over 0.5 PSNR improvement, while previous HSI denoising transformer with GSA only outperforms QRU by only 0.1 PSNR.

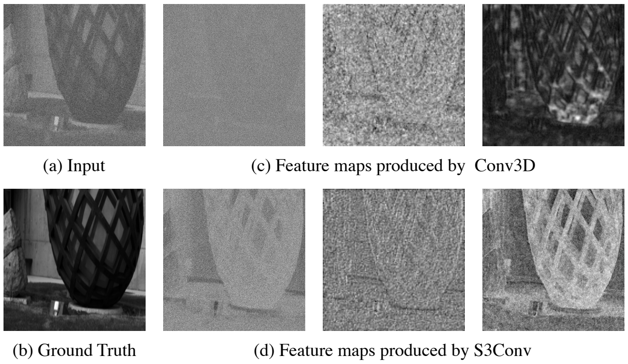
Self-Modulated Feed-Forward Network. The proposed SM-Branch can be used without additional conventional FFN. As shown in Tab. 6, the sole use of SM-Branch also outperforms the conventional FFN, and the combination of them both yields the best results with very few extra parameters. The GDFN [15] developed for RGB restoration performs poorly and might be unsuitable for our model.

Model	Params	PSNR	SAM
FFN	0.49M	41.67	0.050
GDFN [15]	0.49M	37.38	0.094
SM-Branch	0.45M	41.74	0.051
SM-FFN	0.52M	**41.82**	**0.049**

Table 6: Comparison of the existing FFN with our SM-FFN and SM-Branch.

C. More Discussions

Visualization of S3Conv. To demonstrate the effectiveness of our S3Conv. We provide a comparison of the features map between S3Conv and conventional 3D convolution. As shown in Fig. 2, our S3Conv extracts more spatial meaningful features.


(a) Input

(c) Feature maps produced by Conv3D

(b) Ground Truth

(d) Feature maps produced by S3Conv

Figure 2: Comparison of the feature maps extracted by conventional 3D convolution and our S3Conv.

Analysis of SM-FFN. The proposed SM-FFN is designed for strengthening the features with higher activation via a self-modulation operation. The improvement provided by SM-FFN could be intuitively explained by the emphasis on more informative regions that typically have higher activation. In the following, we provide some possible relations between our SM-FFN and the SiLU [5] activation, which might further imply why our SM-FFN works better. Specifically, The SiLU activation is,

$$y = x \odot \mathrm{sigmoid}(x), \tag{1}$$

where x and y are the input and output feature maps. It can be observed that SiLU could be treated as a kind of self-modulation where the modulation weight is computed from the input itself. However, such homogeneous self-modulation might be limited in expressive abilities. Instead, our SM-FFN employs a heterogeneous self-modulation,

$$y = \mathrm{Linear}_1(x) \odot \mathrm{sigmoid}(\mathrm{Linear}_2(x)), \tag{2}$$

where we adopt two extra linear projections to project input x into two different spaces. This removes the restriction of SiLU where the input x should simultaneously play two roles of features and modulation weight. Thus, our SM-FFN can obtain the advantages of SiLU, *e.g.*, training stability and implicit regularization while maintaining more representation capability. Consequently, it leads to better performance than conventional FFN.

D. Extension as Plug-and-Play Prior

Considering the superior performance of our method on the Gaussian denoising task, we demonstrate that HSDT can be used a plug-and-play (PnP) prior [2] to solve general HSI restoration tasks with proximal algorithms, *e.g.*, ADMM and HQS.

Experimental Setup. We adopt PnP-ADMM [8] to extend our method to the tasks of compressive sensing, and super-resolution. To meet the requirements of PnP algorithms, *i.e.*, Gaussian denoiser for continuous noise strengths, we retrain our model, *i.e.*, HSDT-M, with an additional noise level map [16] on simulated Gaussian noise ranged from 0 to 70. We run 40 iterations for compressive sensing and 24 iterations for super-resolution. The hyperparameters of the algorithms are manually tuned to achieve the best performance.

Compressive Sensing. We conduct the simulated experiments on CASSI [13] system. Following [12], the shifting random binary mask [11] is used in our simulation. We provide the results on CAVE `Toy`, which is obtained from [10]. We compare several recent methods, including DPHSIR [8], SCI-TV-FFDNet [12], DeSCI [10], and traditional methods, *i.e.*, 2DTV and 3DTV. The quantitative results are shown in Tab. 1a. It can be seen that our method obtains the best performance with over 1 dB improvement on PSNR. Specifically, the improvement is purely obtained through the superior denoising ability of our model, which means our model can also be integrated into other more advanced PnP methods for further improvement, *e.g.*, [12].

Super-Resolution. We also provide results on the task of HSI super-resolution. Following [8], we first blur the high-resolution HSI via an 8×8 Gaussian blur kernel with $\sigma = 3$, and then downsample the image to obtain the low-resolution HSI. We provide the results on ICVL with a scale factor of 2 and 4. The competing methods include several recently developed methods, *e.g.*, SSPSR [7], Bi3DQRNN [6], and DPHSIR [8] . As shown in Tab. 1b, our method achieves the best performance. In particular, our method only needs the pretrained Gaussian denoising model, which is the same as [8]. The improvement against [8] comes from the better PnP denoising prior, which further demonstrates the stronger denoising ability of our method.

E. More Implementation Details

Setup of the Learning Rate. In this part, we provide more details about the multi-step learning rate scheduler that we used for training our simulated Gaussian and complex denoising models. Specifically, we use a multi-stage training strategy to train the models for Gaussian noise and complex noise. The learning rate is set up as shown in Tab. 3a. We use learning rate warmup to gradually increase the learning rate from 0 to 1×10^{-3} for the first epoch of the second stage.

Details of the Simulated Complex Noise. We follow [14] for constructing simulated complex noise. In details, we consider the non-independent and non-identically distributed (non-i.i.d) Gaussian noise, stripe noise, deadline noise, impulse noise, and the combination of the aforementioned noise (denoted as mixture noise). The details about these five cases of noise are listed as follows,

- **Non-i.i.d noise**. The non-independent and non-identically distributed Gaussian is added to every pixel of each HSI. The noise strength is randomly selected from 10, 30, 50, and 70.
- **Stripe noise**. Stripe noise (5% to 15% percentages of columns) is added to randomly selected one-third of bands. Non-i.i.d. Gaussian noise is added to All bands.
- **Deadline noise**. Deadline noise is added to randomly selected one-third of bands. Non-i.i.d. Gaussian noise is added to All bands.
- **Impulse noise**. Impulse noise with intensity ranging from 10% to 70% is added to randomly selected one-third of bands. Non-i.i.d. Gaussian noise is added to All bands.
- **Mixture noise**. Each band is randomly corrupted by at least one kind of noise mentioned above.

System Configuration. In the main paper, we compare the running time of different methods. All the comparisons are performed with an Nvidia GeForce RTX 3090, and an Intel(R) Core(TM) i9-10850K CPU @ 3.60GHz on Ubuntu 20.04.1 LTS. All the CNN-based methods are implemented and tested with PyTorch 1.7.1. All the optimization-based methods are implemented and tested with Matlab. We test the running time on ICVL with an image size of 512×512 by repeating the test 10 times and averaging the results.

F. Future work.

In this work, we propose a transformer architecture, *i.e.*, HSDT for hyperspectral image denoising. We introduce several effective and generalizable components to better explore the spatial-spectral and global spectral correlations of HSI. Specifically, it is worthwhile to explore the applications of the proposed S3Conv and HSDT for more network architectures and tasks. Furthermore, our learnable queries

Method	PSNR	SSIM
2DTV	25.26	0.863
3DTV	28.46	0.910
DeSCI [10]	26.62	0.912
SCI-TV-FFDNet [12]	29.35	0.925
DPHSIR [8]	30.56	0.945
PnP-HSDT (ours)	**31.64**	**0.948**

(a) Results on the task of compressive sensing.

	2x		4x	
Method	PSNR	SSIM	PSNR	SSIM
Bicubic	35.13	0.9575	35.12	0.954
SSPSR [7]	47.55	0.995	39.19	0.979
Bi-3DQRNN [6]	42.53	0.989	39.56	0.979
DPHSIR [8]	48.75	0.996	40.95	0.980
PnP-HSDT (ours)	**49.76**	**0.996**	**41.56**	**0.982**

(b) Results on the task of super-resolution.

Table 7: Experimental results of our PnP extension on the task of compressive sensing and super-resolution.

Stage 1	Gaussian Noise $\sigma = 50$					
Epoch	0 - 20	20 - 30				
LR	1×10^{-3}	1×10^{-4}				
Stage 2	Gaussian Noise $\sigma = 10, 30, 50, 70$					
Epoch	30 - 45	45 - 55	55 - 60	60 - 65	65 - 75	75 - 80
LR	1×10^{-3}	1×10^{-4}	5×10^{-5}	1×10^{-5}	5×10^{-6}	1×10^{-6}
Stage 3	Complex Noise					
Epoch	80 - 90	90 - 95	95 - 100	100 - 105	105 - 110	
LR	1×10^{-3}	5×10^{-4}	1×10^{-4}	5×10^{-5}	1×10^{-5}	

(a) Our multi-step learning rate scheduler.

System	Ubuntu 20.04.1 LTS
GPU	Nvidia GeForce RTX 3090
CPU	Intel(R) Core(TM) i9-10850K CPU
Framework	PyTorch 1.7.1
Driver	Cuda 11.2
Software	Matlab 2020
Dataset	ICVL
Image Size	512×512
Repeat times	10

(b) System configuration for the speed test.

Table 8: More implementation details. (a) We adopt a multi-stage training strategy with the learning warmup setup for the first epoch. (b) We provide the system configuration as the results of the speed test are strongly correlated with the configuration.

could also be extended to condition on some external information for more explicit guidance. For example, we might be able to inject the Gaussian noise strength into the network with learnable queries, through an embedding layer. This is helpful for a PnP Gaussian denoiser, where the noise strength is known.

G. Broader Impacts

Our work has no ethical issues or broader impacts.

References

[1] Yuanhao Cai, Jing Lin, Xiaowan Hu, Haoqian Wang, Xin Yuan, Yulun Zhang, Radu Timofte, and Luc Van Gool. Mask-guided spectral-wise transformer for efficient hyperspectral image reconstruction. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 17502–17511, 2022. 1, 3

[2] Stanley H Chan, Xiran Wang, and Omar A Elgendy. Plug-and-play admm for image restoration: Fixed-point convergence and applications. *IEEE Trans. Comput. Imaging.*, 3(1):84–98, 2016. 4

[3] Hongyu Chen, Guangyi Yang, and Hongyan Zhang. Hider: A hyperspectral image denoising transformer with spatial–spectral constraints for hybrid noise removal. *IEEE Transactions on Neural Networks and Learning Systems*, 2022. 1, 3

[4] Weisheng Dong, Huan Wang, Fangfang Wu, Guangming Shi, and Xin Li. Deep spatial–spectral representation learning for hyperspectral image denoising. *IEEE Trans. Comput. Imaging.*, 5(4):635–648, 2019. 2, 3

[5] Stefan Elfwing, Eiji Uchibe, and Kenji Doya. Sigmoid-weighted linear units for neural network function approximation in reinforcement learning. *Neural Networks*, 107:3–11, 2018. 3

[6] Ying Fu, Zhiyuan Liang, and Shaodi You. Bidirectional 3d quasi-recurrent neural networkfor hyperspectral image super-resolution. *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.*, 14:2674–2688, 2021. 4, 5

[7] Junjun Jiang, He Sun, Xianming Liu, and Jiayi Ma. Learning spatial-spectral prior for super-resolution of hyperspectral imagery. *IEEE Trans. Comput. Imaging.*, 6:1082–1096, 2020. 4, 5

[8] Zeqiang Lai, Kaixuan Wei, and Ying Fu. Deep plug-and-play prior for hyperspectral image restoration. *Neurocomputing*, 481:281–293, 2022. 4, 5

[9] Miaoyu Li, Ying Fu, and Yulun Zhang. Spatial-spectral transformer for hyperspectral image denoising. *arXiv preprint arXiv:2211.14090*, 2022. 1, 2, 3

[10] Yang Liu, Xin Yuan, Jinli Suo, David J. Brady, and Qionghai Dai. Rank minimization for snapshot compressive imaging. *IEEE Trans. Pattern Anal. Mach. Intell.*, 41(12):2990 – 3006, 2019. 4, 5

[11] Patrick Llull, Xuejun Liao, Xin Yuan, Jianbo Yang, David Kittle, Lawrence Carin, Guillermo Sapiro, and David J Brady. Coded aperture compressive temporal imaging. *Optics express*, 21(9):10526–10545, 2013. 4

[12] Haiquan Qiu, Yao Wang, and Deyu Meng. Effective snapshot compressive-spectral imaging via deep denoising and total variation priors. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 9127–9136, 2021. 4, 5

[13] Ashwin A Wagadarikar, Nikos P Pitsianis, Xiaobai Sun, and David J Brady. Video rate spectral imaging using a coded aperture snapshot spectral imager. *Optics Express*, 17(8):6368–6388, 2009. 4

[14] Kaixuan Wei, Ying Fu, and Hua Huang. 3-d quasi-recurrent neural network for hyperspectral image denoising. *IEEE Trans Neural Netw Learn Syst.*, 32(1):363–375, 2021. 3, 4

[15] 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 *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 5728–5739, 2022. 1, 3

[16] Kai Zhang, Yawei Li, Wangmeng Zuo, Lei Zhang, Luc Van Gool, and Radu Timofte. Plug-and-play image restoration with deep denoiser prior. *IEEE Trans. Pattern Anal. Mach. Intell.*, 44:6360–6376, 2021. 4