Image Smoothing via Multiscale Global Perception

Xuyi He, Yuhui Quan, Yong Xu and Ruotao Xu

***Abstract*—Image smoothing provides a fundamental operation for image processing, with a broad spectrum of applications. It is a challenging task which requires global analysis on image patterns with scale awareness. Existing deep models for image smoothing are insufficiently efficient in global perception and multi-scale processing. This paper proposes a deep model with an efficient multi-scale fusion architecture and a series of global processing blocks. The architecture enhances multi-scale feature flow by incorporating features of different scales into both the encoder and decoder blocks of a U-shape network, with multi-scale feature fusion modules inserted between the encoder and the decoder. The global processing blocks leverage the multi-axis processing mechanism to achieve joint local and global perception. Benefiting from these two key designs, our proposed model enjoys superiority in both smoothing performance and computational complexity, as demonstrated in the experiments on two benchmark datasets.**

***Index Terms*—Image smoothing, Multi-scale analysis, Global perception, Multi-axis processing, Deep learning**

I. Introduction

IMAGE smoothing aims at smoothing out insignificant textures of an image while preserving meaningful structures, with wide applications such as artistic effect creation in image editing, robustness enhancement for downstream vision tasks, image data reduction, noise removal, and blind quality assessment [1]. It is an important topic with continuous active research in recent years [2]–[4].

Scales and edges are two crucial types of clues for image smoothing. Scales allow utilizing the perceptual prior that textures and structures usually lie at finer and coarser scales, respectively. Edges provide intensity and orientation cues to distinguish between textures and structures. The plain Gaussian smoothing failed in awareness of both edges and scales. In the past decades, many studies focused on edge-aware filtering; bilateral filtering [5]–[7], weighted median filtering [8], guided filtering [9], rolling guided filtering [9], and Gaussian adaptive bilateral filtering [10]. These methods, also known as local filtering, employ a sliding window with weights adapted to structural edges for filtering textures out. One of their weakness is the lack of perception on global cues or constraints during smoothing.

Another popular approach to image smoothing is constructing and solving regularization models; see *e.g.* [2], [11]–[14]. One representative work is the relative total variation (RTV) proposed by Xu *et al.* [12] that introduced a weighting scheme to the total variation (TV) regularizer [15] to better distinguish structural edges and strong textures. He *et al.* [2] proposed the bilateral TV that combines TV with bilateral filtering to further improve the identification of weak structures. Compared to local filtering, these regularization model-based methods have a global perception during smoothing as they perform optimization over the whole image. However, the regularizers they employ only operate on local image regions which cannot fully utilize global information of the image.

Non-local methods [16], [17] provide an effective alternative for exploiting global information. Xu *et al.* [16] introduced a pixel-level non-local operator that exploits self-recurrence prior of natural images for smoothing. Xu *et al.* [17] observed that the self-recurrence prior holds for bother structures and textures. Therefore, they exploited the anisotropy of structural edges and the isotropy of texton distribution to design a directional non-local sparsification transform that only sparsifies texture regions rather than structural edges.

The aforementioned methods leverage manually-designed schemes or regularizers, failing to fit complex image patterns with strong semantics. In recent years, due to their hierarchical and non-linear architectures, Deep Neural Networks (DNNs) have shown promising performance in handling semantic complex textures; see *e.g.* [3], [4], [18]–[23]. Zhu [22] utilized two backbones, VDCNN [24] and ResNet [25], to obtain a larger receptive field for smoothing. Feng *et al.* [23] incorporated edge detection and image smoothing into a DNN, using edge maps containing global clues of image structures for guidance. One limitation of this DNN is that its produced edge maps are weakly supervised by the simple Canny edge detector, due to the absence of ground truths in training data.

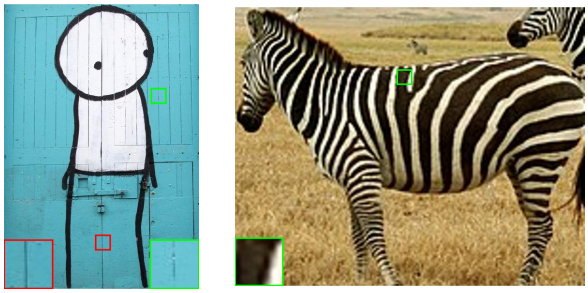
Most existing DNNs for image smoothing focus on enhancing edge awareness. Li *et al.* [3] considered scale awareness by using an invertible DNN with the wavelet transform. The invertible DNN also allows better disentanglement in the deep feature space and a lightweight model. However, the standard coupling layers used by the invertible DNN restrict the DNN's expressivity needed for handling rich textures and structures.

Both edges and scales can be significantly spatially-varying within an image. To achieve effective edge awareness and scale awareness, not only local but also global information on the image should be fully utilized. See Fig. 1 for a demonstration. In Fig. 1(a), the bars on the door are considered as texture since they repeat over the whole door, while the crack of the door is considered as a structural edge. The successful handling of this case requires a DNN to exploit both local and global cues simultaneously. Another example shown in Fig. 1(b) is, the coarse-scale stripes on the zebra's body have strong edges, thus being easily mistaken as structures within a local receptive field. A correct identification of these stripes as

All the authors are with the School of Computer Science and Engineering, South China University of Technology, Guangzhou 510006, China. R. Xu is also with Institute for Super Robotics, Guangzhou 510641, China.

Corresponding authors: Ruotao Xu

This work is supported in part by National Natural Science Foundation of China under Grants 62372186 and 61902130, and in part by Natural Science Foundation of Guangdong Province under Grants 2022A1515011755 and 2023A1515012841, and in part by Fundamental Research Funds for the Central Universities under Grant x2jsD2230220.



(a) (b)

Fig. 1: Two examples for demonstrating the importance of global perception for image smoothing. Red squares: meaningful structures; green squares: repeating textures.

textures requires a large receptive field that covers the whole zebra or at least multiple stripe regions.

However, considering computational efficiency, most existing DNNs (*e.g.* [3], [18]–[23]) are composed of a limited number of convolutional layers, leading to restricted receptive fields. In addition, their multi-scale processing pipelines are usually done by a plain U-shape backbone, which cannot make full use of features from different scales for global analysis. In this work, we present a deep model that combines an efficient multi-scale fusion architecture with a set of global processing blocks, addressing the limitation caused by under-utilizing global information and multi-scale cues in existing DNN-based methods. Sharing a similar spirit with [27]–[30], our multi-scale fusion architecture integrates multi-scale image representation into the encoder and decoder components of a U-shape backbone, while also incorporating multi-scale feature fusion modules between them. The global processing blocks utilize a multi-axis processing mechanism [31] to achieve simultaneous local and global perception for feature extraction. Through these two design elements, our DNN enjoys superior performance over a set of recent methods in terms of smoothing performance and computational complexity, as demonstrated in our experiments on two datasets.

In summary, our main contribution is a new DNN for image smoothing with the following components/merits:

1) an efficient multi-scale fusion architecture;
2) global processing blocks based on multi-axis perception;
3) excellent performance and low computational cost.

II. Methodology

A. Network Architecture for Multi-Scale Processing

Our proposed DNN, called MGPNet (Multi-scale Global Perception Network), takes a textured image as input and predicts a smoothed texture-free version as output. See Fig. 2(a) for an illustration of the architecture of MGPNet. It contains an encoder part and a decoder part, whose blocks are organized in a multi-scale structure inspired by [27], [28]. The encoder part contains three encoder blocks (EBs) which form features from fine scales to coarse scales progressively. Accordingly, the decoder part contains three decoder blocks (DBs) that predict the smoothed image from coarse to fine scales progressively. Between the encoder and decoder parts, multi-scale fusion blocks (MSFBs) are inserted to build up an enhanced pathway of feature flow for multi-scale processing.

Feature encoding All the EBs in MGPNet have the same structure, composed of a 1×1 convolutional layer for feature channel number adjustment and two global processing blocks (GPBs) for global feature extraction, with a skip connection over the GPBs for residual learning. Each EB maps an input feature tensor to a new one of the same spatial size. The input feature tensor of an EB is generated as follows.

Let $\boldsymbol{Y} \in \mathbb{R}^{W\times H\times 3}$ denote the input image and $(\cdot)_{\downarrow m}$ the downsampling operator with factor m. Define the multi-scale versions of $\boldsymbol{Y}$ as follows: $\boldsymbol{Y}_1 = \boldsymbol{Y}$ and $\boldsymbol{Y}_t = \boldsymbol{Y}_{\downarrow 2^{t-1}} \in \mathbb{R}^{\frac{W}{2^{t-1}}\times\frac{H}{2^{t-1}}\times 3}$ for $t > 1$. The EB for the first (finest) scale takes $\boldsymbol{Y}_1$ as input. Regarding the EB for tth scale with $t > 1$, its input is defined by the concatenation of *(i)* the downsampled version of the feature tensor from the EB of the (t-1)th scale and *(ii)* the convolutional feature tensor of $\boldsymbol{Y}_t$ generated by a 3×3 convolutional layer. Formally, the processing done in an EB can be expressed as:

$$\mathrm{EB}_t^{\mathrm{out}} = \begin{cases} \mathrm{EB}_t(\boldsymbol{Y}_t), & \text{if } t = 1, \\ \mathrm{EB}_t(\mathrm{conv}(\boldsymbol{Y}_t), (\mathrm{EB}_{t-1}^{\mathrm{out}})_{\downarrow 2}), & \text{if } t = 2, 3, \end{cases} \tag{1}$$

where "conv" denotes a convolutional layer.

Multi-scale feature fusion There are two MSFBs, each of which fuses features from EBs of distinct scales and feeds them to a DB for improved prediction. Concretely, the MSFBs take feature tensors extracted by all the EBs as input. Before inputting, upsampling or downsampling operations are applied to the feature tensors for size consistency.

Feature decoding The DBs correspond to the EBs of different scales and have the same structure as the EBs, *i.e.*, a 1×1 convolutional layer and two sequential GPBs. Each DB outputs a decoded feature tensor of the same spatial size as its input. The DB for the roughest scale (t=3) directly uses the feature tensor output by the EB at that scale. As for the DB at the tth scale ($t < 3$), its input is defined by the concatenation of *(i)* the upsampled version of the feature tensor output by the DB of (t+1)th scale and *(ii)* the output of the MSFB at the tth scale. The processing in a DB can be expressed as:

$$\mathrm{DB}_t^{\mathrm{out}} = \begin{cases} \mathrm{DB}_t(\mathrm{MSFB}_t^{\mathrm{out}}, (\mathrm{DB}_{t+1}^{\mathrm{out}})_{\uparrow 2}), & \text{if } t = 1, 2, \\ \mathrm{DB}_t(\mathrm{EB}_t^{\mathrm{out}}), & \text{if } t = 3, \end{cases} \tag{2}$$

where $(\cdot)_{\uparrow m}$ is the downsampling operator with factor m.

The decoded feature tensor produced by each DB is fed to a 3×3 convolutional layer, and the result is combined with the input image (downsampled to the corresponding scale) via a skip connection, producing a smoothed image at the corresponding scale. The smoothed image at the finest scale is used as the final prediction, while the ones at the other scales are used for calculating the training loss.

Training loss The MGPNet is trained with a multi-scale loss. Let $\boldsymbol{X}_t$ denote the smoothed image predicted by the DB at the tth scale. Let $\boldsymbol{X}^{\mathrm{gt}}$ denote the ground truth. The training loss is defined as

$$\mathcal{L} = \lambda_1 \|\boldsymbol{X}_1 - \boldsymbol{X}^{\mathrm{gt}}\|_1 + \Sigma_{t=2}^{3} \lambda_t \|\boldsymbol{X}_t - \boldsymbol{X}^{\mathrm{gt}}_{\downarrow 2^{t-1}}\|_1, \tag{3}$$

where the scale-related weights $\lambda_t \in \mathbb{R}_+$ are all set to 1.

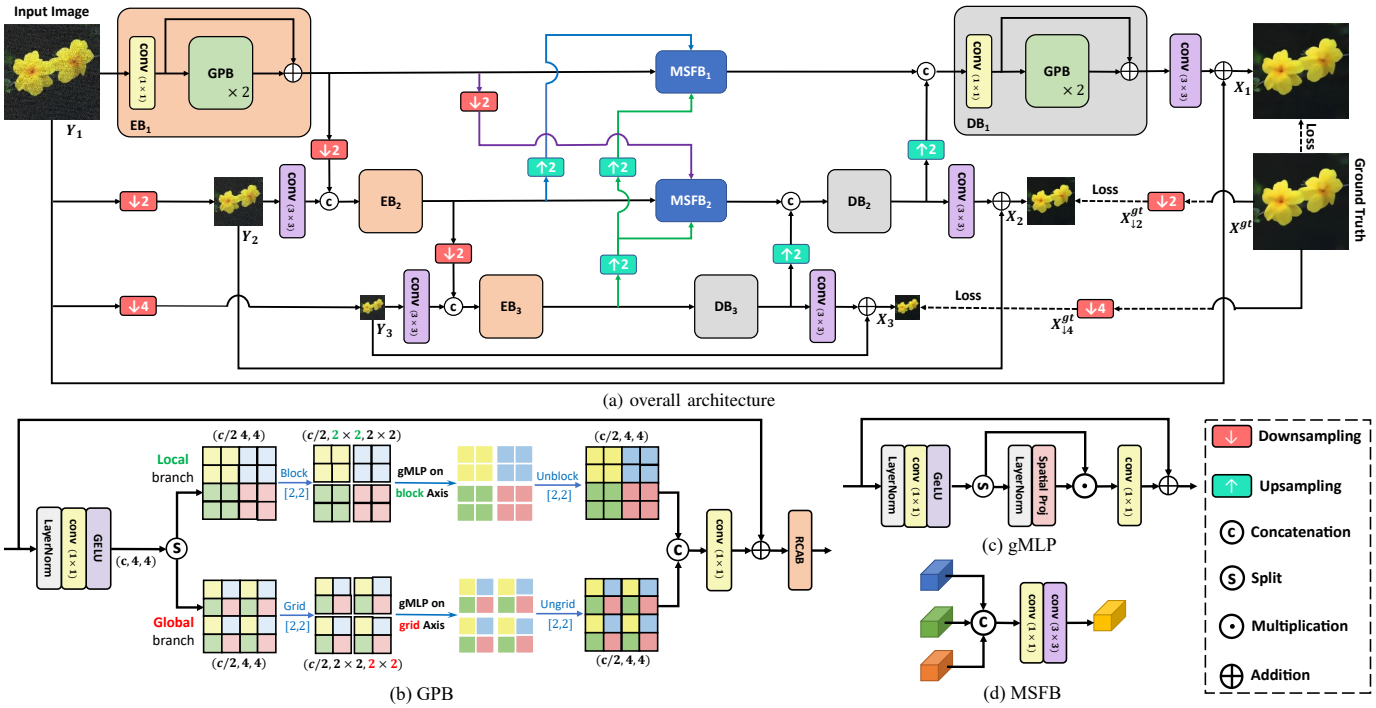


Fig. 2: Architecture of proposed MGPNet for image smoothing. (a) The overall architecture. (b) The structure of GPB. An example is shown with input feature shape $(c, 4, 4)$ and both block size and grid size set to $(2, 2)$. The units of the same color are in the same spatial group of gMLP. Due to space constraints, the details of RCAB [26] are given in the supplementary materials. (c) The structure of gMLP. (d) The structure of MSFB.

B. Details of Key Modules

GPB The GPB aims at extracting features with global perception, which is built upon the multi-axis processing mechanism [32] which arranges spatially-distant features into local paths for global processing. See Fig. 2(b) for its structure. The input feature tensor is first processed by a block consisting of layer normalization (LayerNorm), a 1×1 convolution, and a Gaussian error linear unit (GELU), and then it goes through two parallel branches: the local branch with a local reception field and the global branch with a non-local reception field. This two-branches structure allows each GPB to jointly exploit local and global patterns.

In the local branch, the half head of features with shape $(\frac{c}{2}, h, w)$ is partitioned into several non-overlapping windows of shape (b, b), which is equivalent to reshaping the feature into a tensor of shape $(\frac{c}{2}, b \times b, \frac{h}{b} \times \frac{w}{b})$, called blocking. Here the three dimensions of the transformed feature shape denote channel, block and grid axis, respectively. In the global branch, a (d, d) grid is used to re-organize the other half-head of features into a tensor of shape $(\frac{c}{2}, \frac{h}{d} \times \frac{w}{d}, d \times d)$, called gridding. This operation allows spatially-distant features to be related and processed together. On both branches, the re-organized features are fed to a gated multi-layer perceptron (gMLP) [33] with the structure shown in Fig. 2(c). There is a spatial projection layer in gMLP, which is a linear mapping along the block axis for the local branch and along the grid axis for the global branch. Afterward, unblocking and ungridding are applied on the two branches respectively so as to reshape the features back.

The results from two branches are merged via concatenation followed by a 1×1 convolution. Further, a residual channel attention block (RCAB) [26] is used for channel re-calibration.

MSFB See Fig. 2(d) for the structure of MSFB. Its inputs include three feature tensors from different scales with normalized sizes, which are first concatenated along the channel axis and then go through a 1×1 convolution and a 3×3 convolution. The output feature tensor encodes cues from different scales, which is delivered to the corresponding DB.

III. Experiments

A. Experimental Settings

1) Implementation details: Our MGPNet is implemented in PyTorch and run on an NVIDIA RTX 3090 GPU. The model training is done using the Adam optimizer with batch size of 8 and 3×10^4 iterations. The learning rate is initialized to 2×10^{-4} and decayed by half every 10^4 iterations. In training, images are cropped to 128×128 patches, augmented with horizontal or vertical flipping and rotations of $0°$, $90°$, $180°$ and $270°$. Ours source code will be released after paper's acceptance via github.com/csxyhe/MGPNet.

2) Datasets: The training split of the Structure-Preserving Smoothing (SPS) dataset [23] is used for training, with 1.8×10^5 (2000) samples for training (validation). The performance is evaluated on two test sets: SPS test set with 100 image pairs and Nankai Smoothing (NKS) [16] dataset with 200 image pairs. The characteristics of SPS and NKS datasets differ a lot, *e.g.*, the structure images of SPS mainly contain texture-less natural images, while the ones of NKS mainly contain hand-drawn-like images. This allows a generalization test.

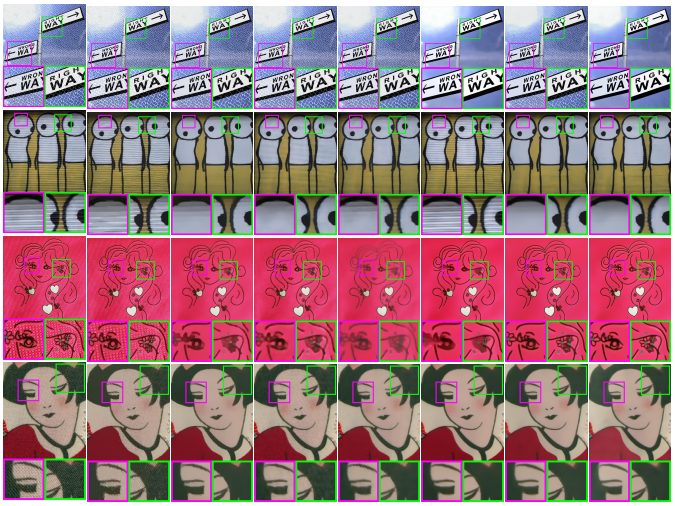


Fig. 3: Visual comparison of results on synthesized images (1st row, from SPS dataset [23]) and natural images (2nd-4th rows, from RTV [12]).

3) Evaluation metrics: Peak signal-to-noise ratio (PSNR) and structural similarity index (SSIM) are as the quantitative metrics. Visual results are provided for subjective inspection.

B. Results and Comparison

Our MGPNet is experimentally compared with (i) two local filtering methods: GF [9] and RGF [34]; (ii) six regularization model-based methods: L0GM [11], RTV [12], SDF [35], PNLS [16], ISO [17] and ILS [13]; (iii) five DNNs-based methods: DEAF [18], DRF [19], ResNet [22], JESSNet [23] and SASNet [3]. For a fair comparison, all the compared DNN models are trained using the same data as ours. The parameters of the local filtering methods and regularization model-based methods are tuned to the universal ones.

See Table I for a quantitative comparison. Our MGPNet outperforms other compared methods in both PSNR and SSIM metrics. It is the top performer in PSNR, with 1.08dB and 0.80dB gain over SASNet on the two test sets, respectively, and also one of the top performers in SSIM. See Fig. 3 for four examples of visual inspection. Our results enjoy better visual quality than that of other compared methods. The third row of Fig. 3 shows a challenging case which requires removing the background texture while maintaining the small structures of the flowers and woman's eyes. L0GM does not smooth the background texture at all, RGF, RTV and ISO over-blur the whole image, and JESSNet fails to maintain the small structures. In comparison, MGPNet wins the trade-off.

Table II compares the number of parameters and the number of floating-point operations (FLOPs) of different models. Our MGPNet enjoys the smallest number of FLOPs, as most of its operations are 1×1 convolution and small-dimension linear operations. This demonstrates that the performance gain of our MGPNet is not from increasing model complexity, but from a more delicate architectural design.

C. Ablation Study

We verify the effectiveness of the multi-scale architecture of MGPNet by *(i)* replacing all EBs and all DBs by the plain ones used in the standard U-Net [36], respectively; and *(ii)* removing the multi-scale design of MSFB. In addition, we verify the effectiveness of GPB by only using its local branch or only using its global branch. We increase the channel number to make the model size of these resulting variants of MGPNet close to the original one. See Table III for the results. We can see that each of our design element has a noticeable performance contribution. For instance, a PSNR gain of 1.12dB is achieved by the multi-scale design of MSFB. Without the local or global branches in the GPB modules, there is a PSNR drop of 0.66dB and 0.30dB, respectively.

TABLE I: Quantitative comparison on two datasets. The best and second-best results are **boldfaced** and underlined, respectively.

Type	Method	SPS		NKS	
		PSNR	SSIM	PSNR	SSIM
Filtering	GF [9]	25.33	0.65	28.15	0.83
Filtering	RGF [34]	25.86	0.64	32.56	0.91
Regularization	L0GM [11]	25.48	0.78	28.32	0.90
Regularization	RTV [12]	26.89	0.82	30.69	0.90
Regularization	SDF [35]	27.06	0.80	33.17	0.89
Regularization	ILS [13]	25.46	0.62	31.50	0.81
Non-local	PNLS [16]	25.43	0.66	33.20	0.92
Non-local	ISO [17]	27.10	0.81	33.25	**0.95**
DNN	DEAF [18]	27.36	0.82	30.45	0.90
DNN	DRF [19]	27.01	0.79	30.02	0.88
DNN	ResNet [22]	29.84	0.88	33.24	0.92
DNN	JESSNet [23]	31.73	0.92	34.24	0.94
DNN	SASNet [3]	33.18	0.94	34.75	**0.95**
DNN	**MGPNet**	**34.26**	**0.95**	**35.55**	**0.95**

TABLE II: Number of parameters and number of FLOPs of different DNNs in processing a 512×384 image. The lowest values are **boldfaced**.

Method	ResNet	JESSNet	SASNet	**MGPNet**
#Params (M)	**1.97**	3.42	2.17	3.72
#FLOPs (G)	387.5	672.0	209.1	**92.1**

TABLE III: Effectiveness of different components of MGPNet on SPS test dataset. The terms C, L, G respectively denote the utilization of the complete GPB, solely the local branch of GPB, and solely the global branch of GPB. The best results are **boldfaced**.

EB	DB	MSFB	GPB C / L / G	PSNR (dB)	#Params (M)
Plain	Plain	w/ MS	C	33.69	3.713
Plain	Proposed	w/ MS	C	33.86	3.719
Proposed	Plain	w/ MS	C	33.37	3.717
Proposed	Proposed	wo/ MS	C	33.14	3.717
Proposed	Proposed	w/ MS	L	33.96	3.696
Proposed	Proposed	w/ MS	G	33.60	3.696
Proposed	Proposed	w/ MS	C	**34.26**	3.723

IV. Conclusion

The paper proposed an efficient DNN for image smoothing, which integrates a multi-scale fusion architecture with multiple global processing blocks. The experimental results have shown that our proposed DNN enjoys not only superior performance in both quantitative and visual comparison, but also high computational efficiency. However, the size of our model is slightly larger compared to other models, though it has a small number of FLOPs. In future, we will investigate models with higher compactness.

REFERENCES

[1] F. Zhang and B. Roysam, "Blind quality metric for multidistortion images based on cartoon and texture decomposition," *IEEE Signal Processing Letters*, vol. 23, no. 9, pp. 1265–1269, 2016.

[2] L. He, Y. Xie, S. Xie, and Z. Chen, "Structure-preserving texture smoothing via scale-aware bilateral total variation," *IEEE Transactions on Circuits and Systems for Video Technology*, 2022.

[3] J. Li, K. Qin, R. Xu, and H. Ji, "Deep scale-aware image smoothing," in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*. IEEE, 2022, pp. 2105–2109.

[4] M. Li, Y. Fu, X. Li, and X. Guo, "Deep flexible structure preserving image smoothing," *Proceedings of the 30th ACM International Conference on Multimedia*, 2022.

[5] C. Tomasi and R. Manduchi, "Bilateral filtering for gray and color images," in *Proceedings of the Sixth international conference on computer vision*. IEEE, 1998, pp. 839–846.

[6] H. Cho, H. Lee, H. Kang, and S. Lee, "Bilateral texture filtering," *ACM Transactions on Graphics*, vol. 33, no. 4, pp. 1–8, 2014.

[7] G. Dong and S. T. Acton, "On the convergence of bilateral filter for edge-preserving image smoothing," *IEEE Signal Processing Letters*, vol. 14, no. 9, pp. 617–620, 2007.

[8] Z. Ma, K. He, Y. Wei, J. Sun, and E. Wu, "Constant time weighted median filtering for stereo matching and beyond," in *Proceedings of the IEEE International Conference on Computer Vision*, 2013, pp. 49–56.

[9] K. He, J. Sun, and X. Tang, "Guided image filtering," in *Proceedings of the European conference on computer vision*. Springer, 2010, pp. 1–14.

[10] B.-H. Chen, Y.-S. Tseng, and J.-L. Yin, "Gaussian-adaptive bilateral filter," *IEEE Signal processing letters*, vol. 27, pp. 1670–1674, 2020.

[11] L. Xu, C. Lu, Y. Xu, and J. Jia, "Image smoothing via l0 gradient minimization," in *Proceedings of the 2011 SIGGRAPH Asia conference*, 2011, pp. 1–12.

[12] L. Xu, Q. Yan, Y. Xia, and J. Jia, "Structure extraction from texture via relative total variation," *ACM transactions on graphics*, vol. 31, no. 6, pp. 1–10, 2012.

[13] W. Liu, P. Zhang, X. Huang, J. Yang, C. Shen, and I. Reid, "Real-time image smoothing via iterative least squares," *ACM Transactions on Graphics*, vol. 39, no. 3, pp. 1–24, 2020.

[14] H. Eun and C. Kim, "Superpixel-guided adaptive image smoothing," *IEEE Signal Processing Letters*, vol. 23, no. 12, pp. 1887–1891, 2016.

[15] L. I. Rudin, S. Osher, and E. Fatemi, "Nonlinear total variation based noise removal algorithms," *Physica D: nonlinear phenomena*, vol. 60, no. 1-4, pp. 259–268, 1992.

[16] J. Xu, Z.-A. Liu, Y.-K. Hou, X.-T. Zhen, L. Shao, and M.-M. Cheng, "Pixel-level non-local image smoothing with objective evaluation," *IEEE Transactions on Multimedia*, vol. 23, pp. 4065–4078, 2020.

[17] R. Xu, Y. Xu, and Y. Quan, "Structure-texture image decomposition using discriminative patch recurrence," *IEEE Transactions on Image Processing*, vol. 30, pp. 1542–1555, 2020.

[18] L. Xu, J. Ren, Q. Yan, R. Liao, and J. Jia, "Deep edge-aware filters," in *Proceedings of the International Conference on Machine Learning*. PMLR, 2015, pp. 1669–1678.

[19] S. Liu, J. Pan, and M.-H. Yang, "Learning recursive filters for low-level vision via a hybrid neural network," in *Proceedings of the European conference on computer vision*. Springer, 2016, pp. 560–576.

[20] Q. Fan, J. Yang, G. Hua, B. Chen, and D. Wipf, "A generic deep architecture for single image reflection removal and image smoothing," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 3238–3247.

[21] Q. Chen, J. Xu, and V. Koltun, "Fast image processing with fully-convolutional networks," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 2497–2506.

[22] F. Zhu, Z. Liang, X. Jia, L. Zhang, and Y. Yu, "A benchmark for edge-preserving image smoothing," *IEEE Transactions on Image Processing*, vol. 28, no. 7, pp. 3556–3570, 2019.

[23] Y. Feng, S. Deng, X. Yan, X. Yang, M. Wei, and L. Liu, "Easy2hard: Learning to solve the intractables from a synthetic dataset for structure-preserving image smoothing," *IEEE Transactions on Neural Networks and Learning Systems*, 2021.

[24] A. Conneau, H. Schwenk, L. Barrault, and Y. Lecun, "Very deep convolutional networks for text classification," *arXiv preprint arXiv:1606.01781*, 2016.

[25] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[26] S. Woo, J. Park, J.-Y. Lee, and I. S. Kweon, "Cbam: Convolutional block attention module," in *Proceedings of the European conference on computer vision*, 2018, pp. 3–19.

[27] S.-J. Cho, S.-W. Ji, J.-P. Hong, S.-W. Jung, and S.-J. Ko, "Rethinking coarse-to-fine approach in single image deblurring," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, October 2021, pp. 4641–4650.

[28] C. Guo, C. Li, J. Guo, R. Cong, H. Fu, and P. Han, "Hierarchical features driven residual learning for depth map super-resolution," *IEEE Transactions on Image Processing*, vol. 28, no. 5, pp. 2545–2557, 2019.

[29] K. Jiang, Z. Wang, P. Yi, C. Chen, B. Huang, Y. Luo, J. Ma, and J. Jiang, "Multi-scale progressive fusion network for single image deraining," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 8346–8355.

[30] Y. Xiao, X. Su, Q. Yuan, D. Liu, H. Shen, and L. Zhang, "Satellite video super-resolution via multiscale deformable convolution alignment and temporal grouping projection," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 60, pp. 1–19, 2021.

[31] Z. Tu, H. Talebi, H. Zhang, F. Yang, P. Milanfar, A. Bovik, and Y. Li, "Maxim: Multi-axis mlp for image processing," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 5769–5780.

[32] L. Zhao, Z. Zhang, T. Chen, D. Metaxas, and H. Zhang, "Improved transformer for high-resolution gans," *Advances in Neural Information Processing Systems*, vol. 34, pp. 18 367–18 380, 2021.

[33] H. Liu, Z. Dai, D. So, and Q. V. Le, "Pay attention to mlps," *Advances in Neural Information Processing Systems*, vol. 34, pp. 9204–9215, 2021.

[34] Q. Zhang, X. Shen, L. Xu, and J. Jia, "Rolling guidance filter," in *Proceedings of the European conference on computer vision*. Springer, 2014, pp. 815–830.

[35] B. Ham, M. Cho, and J. Ponce, "Robust guided image filtering using nonconvex potentials," *IEEE transactions on pattern analysis and machine intelligence*, vol. 40, no. 1, pp. 192–207, 2017.

[36] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *Proceedings of the Medical Image Computing and Computer-Assisted Intervention*. Springer, 2015, pp. 234–241.