
Adaptive Window Pruning for Efficient Local Motion Deblurring

Haoying Li^{1,2}, Jixin Zhao¹, Shangchen Zhou¹, Huajun Feng², Chongyi Li^{1*}, Chen Change Loy¹

¹S-Lab, Nanyang Technological University

²State Key Laboratory of Modern Optical Instrumentation, Zhejiang University
{n2207928h, ZHA0038, s200094, chongyi.li, ccloy}@ntu.edu.sg fenghj@zju.edu.cn
https://leiali.github.io/LMD-ViT_webpage/index.html

Abstract

Local motion blur commonly occurs in real-world photography due to the mixing between moving objects and stationary backgrounds during exposure. Existing image deblurring methods predominantly focus on global deblurring, inadvertently affecting the sharpness of backgrounds in locally blurred images and wasting unnecessary computation on sharp pixels, especially for high-resolution images. This paper aims to adaptively and efficiently restore high-resolution locally blurred images. We propose a local motion deblurring vision Transformer (LMD-ViT) built on adaptive window pruning Transformer blocks (AdaWPT). To focus deblurring on local regions and reduce computation, AdaWPT prunes unnecessary windows, only allowing the active windows to be involved in the deblurring processes. The pruning operation relies on the blurriness confidence predicted by a confidence predictor that is trained end-to-end using a reconstruction loss with Gumbel-Softmax re-parameterization and a pruning loss guided by annotated blur masks. Our method removes local motion blur effectively without distorting sharp regions, demonstrated by its exceptional perceptual and quantitative improvements (+0.24dB) compared to state-of-the-art methods. In addition, our approach substantially reduces FLOPs by 66% and achieves more than a twofold increase in inference speed compared to Transformer-based deblurring methods. We will make our code and annotated blur masks publicly available.

1 Introduction

Contrary to global motion blur, which typically affects an entire image [31], local motion blur is confined to specific regions within an image. Such local blur is generally the result of object movements captured by stationary cameras [10, 19]. Attempting to apply global deblurring techniques to images featuring local motion blur can inevitably introduce unwanted distortions in regions that were originally sharp, as illustrated in Figure 1. Moreover, the processing of sharp regions, which is not required in this context, leads to unnecessary computational expenditure. This wastage becomes particularly noticeable when dealing with high-resolution inputs.

Existing local motion deblurring methods, such as LBAG [10], address the issue by detecting local blur regions for targeted processing. While LBAG uses a gate structure to mitigate the deblurring impact on non-blurred regions, preventing unwanted distortion in the background and static objects, it still involves unnecessary computations as the entire image is processed by the network. In addition, the method’s reliance on a Convolutional Neural Network (CNN) architecture leads to limitations, as the interactions between the image and convolution kernels are content-independent and ill-suited for

*Corresponding author

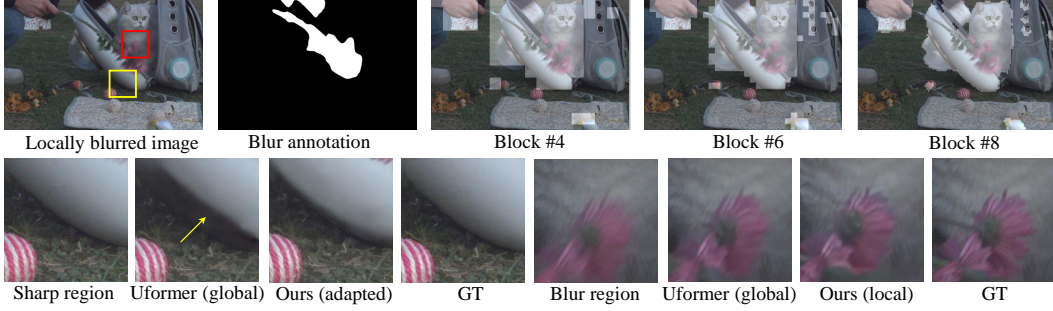


Figure 1: We introduce LMD-ViT, a Transformer-based local motion deblurring method with an adaptive window pruning mechanism. We prune unnecessary windows based on the predicted blurriness confidence supervised by our blur region annotation. In this process, the feature maps are pruned at varying levels of granularity within blocks of different resolutions (as depicted in the 3rd~5th visualizations in Row 1). Unlike global deblurring methods that modify global regions [24, 28], LMD-ViT performs dense computing only on the active windows of blurry regions. Consequently, local blurs are efficiently removed without distorting sharp regions, as shown in Row 2.

modeling long-range dependencies. The Transformer architecture [22], which excels at long-range pixel interactions, has been successfully applied in several image restoration problems [11, 5, 24, 28]. However, Transformers tend to require substantial memory and extend inference time, especially when processing high-resolution images. To mitigate these computational demands, strategies such as window-division [24, 26] and token-reducing [1, 12, 15, 17, 27] have been proposed.

Drawing inspiration from preceding work, in this paper, we propose a U-shaped local motion deblurring vision Transformer (LMD-ViT) with adaptive window pruning Transformer blocks (AdaWPT) as its core component. Our model adaptively processes local blur regions and yields direct speedup, surpassing prior state-of-the-art methods in both deblurring performance and efficiency. The goal of LMD-ViT is to focus on locally blurred regions rather than global regions, which is made possible by removing windows unrelated to blurred areas. Specifically, we first train a confidence predictor which is able to automatically predict the confidence of blurriness of feature maps. It is trained end-to-end by a reconstruction loss with Gumbel-Softmax re-parameterization, and a pruning loss guided by our elaborately annotated local blur masks. We then design a decision layer that provides binary decision maps in which “1” represents the kept tokens in blur-related regions that require processing during inference while “0” represents the abandoned tokens in the other regions that can be removed. We also propose a window pruning strategy with Transformer layers. In detail, we apply window-based multi-head self-attention (W-MSA) and window-based feed-forward layers rather than enforcing these Transformer layers globally. Only the selected windows are forwarded to these window-based Transformer layers, preventing unnecessary distortion of sharp regions while also reducing computations. To further enhance content interactions, AdaWTP employs shifted window mechanism [11, 13] and position embeddings [24] among the Transformer layers. Furthermore, we insert AdaWTP in LMD-ViT under different scales and receptive fields. Therefore, AdaWTP conducts coarse pruning of windows in low-resolution layers and more refined pruning in high-resolution layers, as shown in Figure 1, which achieves a balance between computational complexity and accuracy.

To summarize, our main contributions are 1) a novel window-based Transformer framework for local motion deblurring, LMD-ViT, focusing computation on localized regions affected by blur and achieving efficient and effective blur reduction without causing unnecessary distortion to sharp regions; 2) an adaptive window pruning Transformer block (AdaWPT), which prunes unnecessary windows according to a decision layer as well as a confidence predictor, and speeds up Transformer layers by window pruning strategies; 3) carefully annotated local blur masks for ReLoBlur dataset [10], which improve the performance of local deblurring methods.

2 Related work

Single image deep motion deblurring. The task of deep motion deblurring for single images originated from global deblurring [31, 18, 20, 29, 30, 32]. Pioneering deep global motion deblurring works utilize CNNs as basic layers and achieve promising improvements in image quality. Among them,

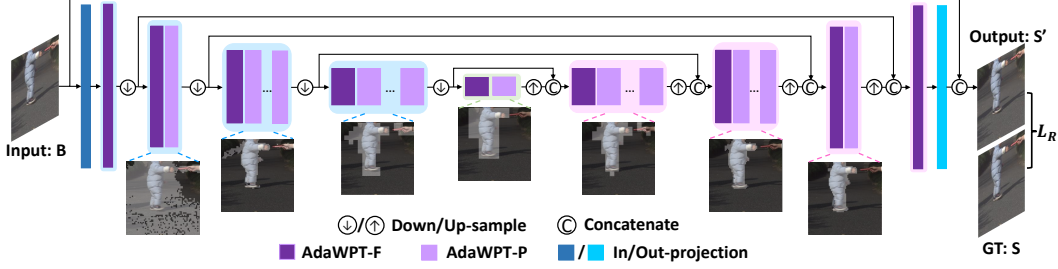


Figure 2: The architecture of LMD-ViT. LMD-ViT is built on a U-shape encoder-decoder structure with AdaWPT blocks. Each AdaWPT block can be further divided into an AdaWPT-F block and several AdaWPT-P blocks. The images with masks below each block depict visualizations of the window pruning effect. The masks indicate the unpruned windows on locally blurred patches and exhibit different levels of pruning granularity.

DeepDeblur [16], a multi-scale convolutional neural network, performs residual blocks to increase convergence speed. DeblurGAN [8] and DeblurGAN-v2 [9] introduce GANs and a perceptual loss to improve subjective quality. HINet [2] applies Instance Normalization to boost performance. Recently, a CNN-based local motion deblurring method, LBAG [10], bridges the gap between global and local motion deblurring by inserting gate modules at the end of MIMO-UNet architecture [3]. It predicts differentiable blur masks to reduce sharp backgrounds from modifications and guide the network to deblur locally. Although the performance is significantly improved, CNN-based methods suffer from the content-independent interactions between images and convolution kernels, as well as the limitations of long-range dependency modeling.

Given the Vision Transformer’s (ViT) [4] ability to capture long-range dependencies, its application to global deblurring tasks has seen a surge of interest. For example, Uformer [24] employs window-based self-attention with a learnable multi-scale restoration modulator to capture both local and global dependencies. Restormer [28] utilizes multi-head attention and a feed-forward network to achieve long-range pixel interactions. In this paper, we build a Transformer-based local motion deblurring framework, LMD-ViT, that adaptively selects windows relevant to blurry regions for window-based self-attention and feed-forward operations, simultaneously benefiting from long-range modeling.

Vision Transformer acceleration. Transformers have proven valuable in deblurring tasks, yet their direct application in local motion deblurring for high-resolution images presents challenges concerning computational efficiency. To solve the heavy computation problem of global self-attention in Transformers, researchers have presented several techniques. For example, Wang et al. adopted pyramid structures and spatial-reduction attention [23] in image classification, object detection, and segmentation tasks. Some methods partition image features into different windows and perform self-attention on local windows [21, 26, 24] for image restoration tasks. Some image classification methods gradually reduce tokens in processing by token-halving [15, 17, 27] or token-merging [12, 1]. Inspired by these advancements, we develop adaptive window pruning blocks (AdaWPT) to eliminate unnecessary tokens and focus deblurring only on blurred regions, which improves image quality and enables inference speed-up without compromising sharp regions.

3 Methodology

3.1 Model architecture

The architecture of our local motion deblurring vision Transformer (LMD-ViT) is shown in Figure 2, which is a U-shaped multi-scale network with an encoder, a bottleneck stage, and a decoder stage with skip connections. An in-projection/out-projection layer is placed at the beginning/end of the network to extract RGB images to feature maps or convert feature maps to RGB images. The encoder, bottleneck, and decoder include a series of adaptive window-token pruning Transformer blocks (AdaWPT) and down-sampling/up-sampling layers. As a key component, AdaWPT removes local blurs by a window pruning strategy with a confidence predictor, a decision layer, and several Transformer layers. It is trained with a reconstruction loss and a pruning loss (as illustrated in Figure 3) constrained by our carefully annotated blur masks. We detail the architectures and model hyper-parameters of LMD-ViT in the supplementary material.

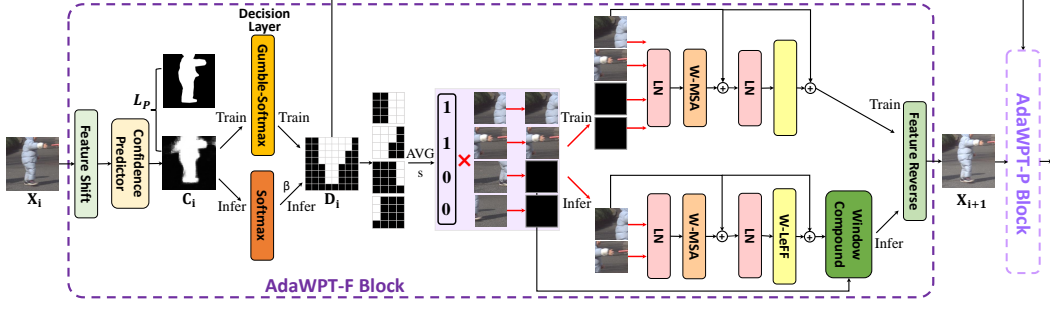


Figure 3: Structure of AdaWPT in training and inference phase. There are two kinds of AdaWPT: AdaWPT-F and AdaWPT-P. AdaWPT-F predicts the confidence of blurriness (“C”) and pruning decisions (“D”), and AdaWPT-P follows the pruning decisions. Both AdaWPT-F and AdaWPT-P prune windows in Transformer layers. “X” denotes the feature map.

3.2 Adaptive window pruning Transformer (AdaWPT)

As shown in Figure 3, an adaptive window pruning Transformer block (AdaWPT) includes a First-AdaWPT block (AdaWPT-F) and several Post-AdaWPT blocks (AdaWPT-P). During training, each AdaWPT-F comprises a confidence predictor, a decision layer, a feature shift/reverse block, and several Transformer layers such as window-based multi-head self-attention (W-MSA) [24], window-based locally-enhanced feed-forward layer (W-LeFF), and layer normalization (LN). A window compound layer is added in inference. AdaWPT-F is responsible for predicting the confidence of blurriness by a confidence predictor, generating pruning decisions from a decision layer and pruning windows. In order to save computational resources, AdaWPT-P follows the decisions provided by AdaWPT-F to prune without producing the confidence of blurriness and decisions again. Therefore, AdaWPT-P does not contain a confidence predictor or a decision layer. In both kinds of AdaWPT, only the unpruned windows are fed into the window-based Transformer layers, significantly reducing computational costs and keeping sharp regions undistorted. Besides, a feature shift/reverse block is inserted before/after pruning to promote feature interactions. We detail the modules in AdaWPT of the following paragraphs.

3.2.1 Confidence predictor

The confidence predictor predicts the confidence of blurriness for the input features $\mathbf{X}_i \in \mathbb{R}^n$. Tokens with higher confidence are more likely to be kept and others are removed. Following Rao et al. [17], the confidence predictor employs MLP layers to produce feature embeddings e and predict the confidence map \mathbf{C} using Softmax:

$$\mathbf{C}_i = \text{Softmax}(e_i), e_i = \text{Concat}\left(\text{MLP}(\text{MLP}(\mathbf{X}_i), \frac{\sum_{j=n}^N \mathbf{D}_j \cdot \text{MLP}(\mathbf{X}_{i_j})}{\sum_{j=n}^N \mathbf{D}_j}\right), i \in \mathbb{N}_+ \quad (1)$$

where \mathbf{D} , initialized using \mathbf{I} , is the one-hot decision map to prune windows, which will be introduced in Section 3.2.2. The confidence predictor is trained using an end-to-end reconstruction loss (see Section 3.4) with Gumbel-Softmax parameterization (see Section 3.2.2), together with a pruning loss (see Section 3.4) guided by our blurry mask annotation (see Section 3.3).

3.2.2 Decision layer

The decision layer samples from the blurriness confidence map to generate binary pruning decisions \mathbf{D} , in which “1” represents tokens to be kept while “0” represents tokens to be abandoned. Although our goal is to perform window pruning, it is not practical to remove the zero tokens directly, for the absence of tokens halts backward propagation, and the different removal instances make parallel computing impossible in end-to-end training. To overcome this issue, we design the decision layer for training and testing, respectively.

In training, we apply the Gumbel-Softmax re-parameterization [6] as the decision layer, since it assures the gradients to flow through the network when sampling the training decision map \mathbf{D}^{tr} from the training confidence map \mathbf{C}^{tr} :

$$\mathbf{D}_i^{\text{tr}}(x, y) = \text{Gumbel-Softmax}(\mathbf{C}^{\text{tr}}(x, y)), \quad (2)$$

where (x, y) represents the coordinate of each window.

In testing, we apply Softmax with a constant threshold β as the decision layer:

$$\mathbf{D}_i^{\text{te}}(x, y) = \begin{cases} 0, & \text{H}(\text{Softmax}(\mathbf{C}_i^{\text{te}}(x, y))) < \beta \\ 1, & \text{H}(\text{Softmax}(\mathbf{C}_i^{\text{te}}(x, y))) \geq \beta, \end{cases} \quad (3)$$

where H functions to sparse the output of Softmax to 1/0 when it is larger/fewer than β . The abandoned windows, that is, windows irrelevant to local blurs, are further set to zero by

$$\mathbf{X}'_i = \mathbf{D}_i \cdot \mathbf{X}_i. \quad (4)$$

3.2.3 Efficient Transformer layers with window pruning strategy

Considering the quadratic computation costs with respect to a large number of tokens of high-resolution images, we employ Transformer layers in non-overlapping windows rather than in a global manner to accelerate training and inference. Specifically, we choose W-MSA layer [24] for the self-attention (SA) operation. W-MSA performs SA on the flattened features in each window and reduces the computational complexity from $O(H^2W^2C)$ to $O(\frac{HW}{M^2}M^4C) = O(M^2HWC)$, where M is the window size. For the feed-forward structure, we develop a window-based locally-enhanced feed-forward layer (W-LeFF) as an alternative to LeFF [24] which modifies global tokens. W-LeFF uses a 3×3 convolution with stride 1 and reflected padding 1 in independent windows. The reflected padding ensures the continuity at the window borders and obtains almost the same performance as LeFF [24] (we compare the performance of LeFF and W-LeFF in the supplementary material).

To enable parallel computing, we regard each window as a token group and prune based on windows. Each window owns a 1/0 decision, which is calculated by average pooling the decision map with a threshold $s = 0.5$. Windows with an average $\geq s$ are regarded as the kept windows and the others as the abandoned windows. To promote content interactions among non-overlapping windows, we also apply relative position encoding [24] in the attention module and shift/reverse the windows by half the window size at the beginning/end of the AdaWTP, which improves deblurring performance.

To accomplish both differentiable training and fast inference, we propose different pruning strategies with W-MSA and W-LeFF in training and testing, respectively. In training, to ensure back-propagation, all the windows including the abandoned windows go through W-MSA, W-LeFF, and LN sequentially to generate training features $\mathbf{X}_{i+1}^{\text{tr}}$:

$$\mathbf{X}_{i+1}^{\text{tr}'} = \text{W-MSA}(\text{LN}(\mathbf{X}_i^{\text{tr}'})) + \mathbf{X}_i^{\text{tr}'}, \quad \mathbf{X}_{i+1}^{\text{tr}} = \text{W-LeFF}(\text{LN}(\mathbf{X}_{i+1}^{\text{tr}'})) + \mathbf{X}_{i+1}^{\text{tr}'}. \quad (5)$$

In testing, only the kept windows are processed by Transformer layers, which release a great number of unnecessary tokens to deblur. To enable future stages to perform global operations, we mend the abandoned windows to their original locations to compound a complete testing feature map $\mathbf{X}_{i+1}^{\text{te}}$:

$$\mathbf{X}_{i+1}^{\text{te}'} = \text{W-MSA}(\text{LN}(\mathbf{X}_i^{\text{te}'} \geq s)) + \mathbf{X}_i^{\text{te}'}, \quad \mathbf{X}_{i+1}^{\text{te}} = \text{W-LeFF}(\text{LN}(\mathbf{X}_{i+1}^{\text{te}'} \geq s)) + \mathbf{X}_{i+1}^{\text{te}'}. \quad (6)$$

3.3 Blur region annotation

To obtain the ground-truth local blur mask for supervised training of the confidence predictor, we carefully annotate the binary local blur masks of the ReLoBlur dataset [10]. We mark the blurred regions with pixel value 1 and others with 0, as shown in the second image in Figure 1. With the help of HRNet18-OCR64 model integrated into *EISeg* software², the blur-sharp regions can be manually segmented, without holes or stripes in each region. Unlike the blur masks marked by LBFMG [10], our manually annotated blur masks are more in line with human visual standards with fewer errors.

3.4 Loss functions

To guide adaptive window pruning and locally deblurring, we propose a pruning loss and combine it with a weighted reconstruction loss to form the total loss:

$$\mathcal{L} = \mathcal{L}_{\mathcal{P}} + \mathcal{L}_{\mathcal{R}}. \quad (7)$$

²EISeg is provided by <https://github.com/PaddlePaddle/PaddleSeg/tree/release/2.7/EISeg>

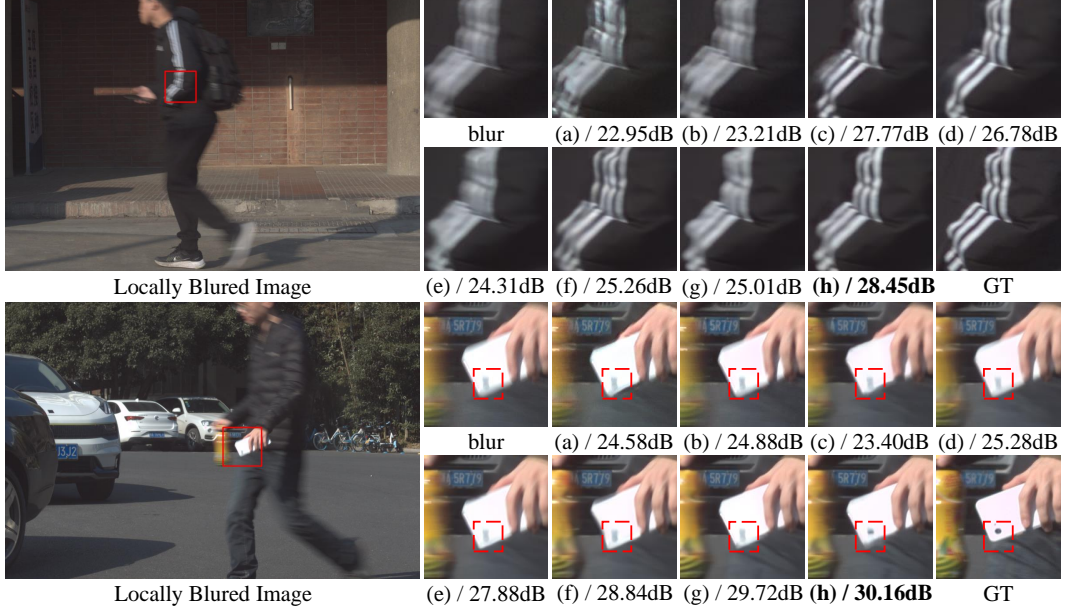


Figure 4: Visual comparisons of state-of-the-art methods for local motion deblurring. (a) DeepDeblur [16]; (b) DeblurGAN_v2 [9]; (c) HINet [2]; (d) MIMO-UNet [3]; (e) LBAG [10]; (f) Restormer [28]; (g) Uformer [24]; (h) LMD-ViT (ours).

Pruning loss. We propose a pruning loss to constrain the blurriness confidence prediction:

$$\mathcal{L}_{\mathcal{P}} = \lambda_0 \sum_{i=1}^n \text{Cross-Entropy}(\mathbf{C}_i, \text{Down-Sample}^i(\mathbf{M})), \quad (8)$$

where $\lambda_0=0.01$, and the blur mask \mathbf{M} is down-sampled to match the resolution of the confidence maps, and i indicates a confidence predictor’s location.

Reconstruction loss. To focus training on local blurred regions, following [10], we apply weights w on the reconstruction loss, which is a combination of L_1 loss, SSIM loss, and FFT loss:

$$\begin{aligned} \mathcal{L}_{\mathcal{R}} &= w\mathcal{L}'_{\mathcal{R}}(\mathbf{M} \cdot \mathbf{S}', \mathbf{M} \cdot \mathbf{S}) + (1 - w)\mathcal{L}'_{\mathcal{R}}((1 - \mathbf{M}) \cdot \mathbf{S}', (1 - \mathbf{M}) \cdot \mathbf{S}), \\ \mathcal{L}'_{\mathcal{R}} &= L_1(\mathbf{S}', \mathbf{S}) + \lambda_2 \text{SSIM}(\mathbf{S}', \mathbf{S}) + \lambda_3 \text{FFT}(\mathbf{S}', \mathbf{S}), \end{aligned} \quad (9)$$

where S , and S' denote the sharp ground truth, and the deblurred output. $w = 0.8$ is the weight, and $\lambda_1=\lambda_2=1.0$, $\lambda_3=0.1$.

4 Experiments and analyses

4.1 Experimental settings

We train LMD-ViT using AdamW optimizer [7] with the momentum terms of (0.9, 0.999), a batch size of 12, and an initial learning rate of 2×10^{-4} that is updated every 2k steps by a cosine annealing schedule [14]. We set the window size of AdaWPT to 8×8 , and the initial embedded dim to 32 which is doubled after passing each down-sampling layer. LMD-ViT is trained on the GoPro dataset [16] and ReLoBlur dataset [10] together. While sampling the training data, we employ a blur-aware patch crop strategy [10] that samples 50% of the 512×512 training data to be blur regions and 50% to be random regions, with the assistance of our blur mask annotations. For a fair comparison, we train the baseline methods using the same datasets and cropping strategy on 4 Nvidia A100 GPUs. The model configurations of the compared deblurring methods follow their origin settings.

We evaluate our proposed LMD-ViT and baseline methods on the ReLoBlur testing dataset [10] with the full image size of 2152×1436 . In addition to the commonly-used PSNR and SSIM [25] metrics, we follow the approach of [10] and calculate weighted PSNR and weighted SSIM specifically for the

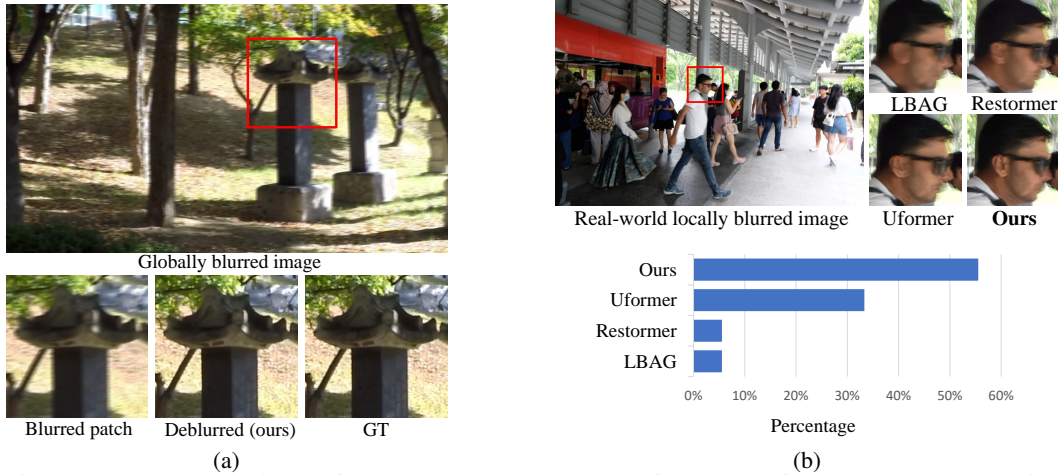


Figure 5: Global deblurring performance (a), and the results of user study for real-world local motion deblurring (b). The vertical axis indicates the percentage of favoring each method.

Table 1: Quantitative comparisons of local deblurring methods. “PSNR_w”, “SSIM_w”, “Time” and “Params” denote weighted PSNR, weighted SSIM, inference time, and model parameters respectively. We bold the best result under each evaluation metric.

Categories	Methods	↑PSNR	↑SSIM	↑PSNR _w	↑SSIM _w	Time	Params	FLOPs
CNNs	DeepDeblur [16]	33.06	0.8935	28.05	0.8404	0.50s	11.72M	17.133T
	DeblurGAN-v2 [9]	33.40	0.9021	28.28	0.8502	0.07s	5.076M	0.989T
	HINet [2]	34.16	0.9123	28.53	0.8617	0.31s	88.67M	8.696T
	MIMO-UNet [3]	34.79	0.9257	29.22	0.8804	0.51s	16.11M	7.850T
	LBAG [10]	34.90	0.9262	29.60	0.8866	0.51s	16.11M	7.852T
Transformers	Restormer [28]	34.92	0.9265	29.47	0.8811	3.72s	26.13M	6.741T
	Uformer-B [24]	35.19	0.9265	30.22	0.8911	1.31s	50.88M	4.375T
	LMD-ViT	35.43	0.9288	30.29	0.8934	0.56s	54.50M	1.485T

blurred regions. This allows us to better assess the local deblurring performance. To measure model efficiency, we report the inference time per image, the number of floating point operations per second (FLOPs) per image, and the model parameters. We provide the evaluation results in the following sections and the supplementary material.

4.2 Experimental results

Evaluations on public datasets. We first compare the proposed LMD-ViT with both CNN-based methods [16, 9, 2, 3, 10] and Transformer-based methods [28, 24] on the ReLoBlur dataset [10] for local motion deblurring. As depicted in Figure 1 and Figure 4, LMD-ViT exhibits superior performance compared to other state-of-the-art methods, producing clearer outputs with enhanced details. Notably, the white stripes on the student’s suit and the mobile phone show significant blur reduction without artifacts and closely resemble the ground truth. We notice that the number of preserved windows slightly exceeds the number of windows in the annotated blurry areas. This is to ensure that the preserved windows effectively cover as much of the blurry area as possible. Quantitative evaluation results are presented in Table 1, which bolds the best local deblurring performance. Compared to CNN-based methods, our proposed LMD-ViT achieves an improvement of 0.53 dB in PSNR and 0.69 dB in weighted PSNR, while maintaining a comparable or even faster inference speed. When compared to Transformer-based methods, LMD-ViT demonstrates significant reductions (-66%) in FLOPs and inference time without sacrificing performance (PSNR +0.24dB), thanks to our adaptive window pruning modules.

Additionally, our proposed LMD-ViT could deblur globally. We evaluate LMD-ViT’s global deblurring performance on the GoPro dataset [16]. As shown in Figure 5(a), our proposed LMD-ViT could remove global blurriness and restore the sharp textures obviously. We also observed that with a

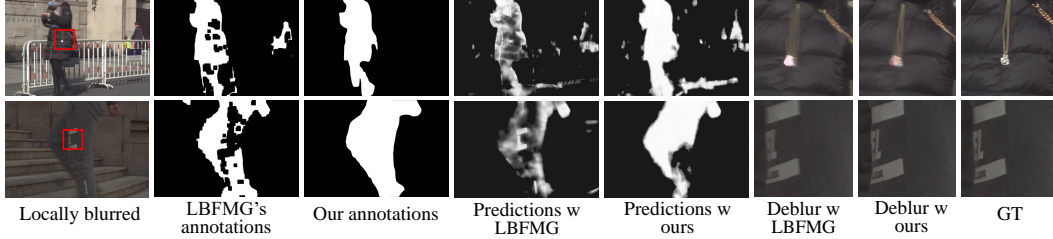


Figure 6: Visual comparisons of our blur mask annotations and LBFMG’s blur masks [10].

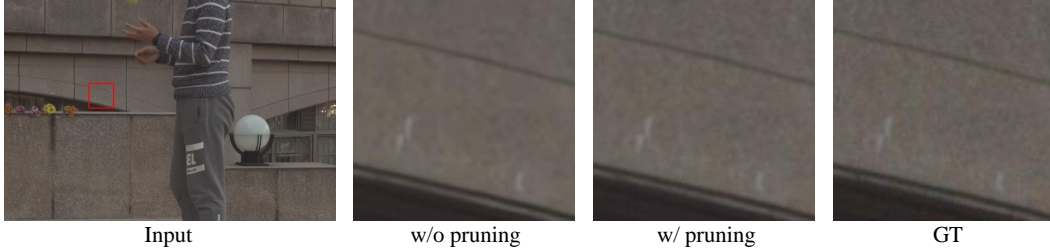


Figure 7: Evaluations for sharp region quality preservation with or without the pruning strategy. Without pruning, the network modifies global pixels and blurs the sharp backgrounds. In contrast, with pruning, sharp backgrounds are preserved well.

globally blurry input, LMD-ViT does not prune windows because the confidence of blurriness of all regions is nearly 100%. We will explain this in detail in the supplementary material.

User study on real-world photos. To validate the effectiveness of our proposed model in real-world locally blurred images, we capture 18 locally blurred RGB images with a resolution of 6000×4000 . We conduct a comparison between our method and the top 3 methods listed in Table 1 on these images. Since the ground truths for these blurred images are not available, we conduct a user study involving 30 participants who are passionate about photography. Each participant is presented with randomly selected deblurred images and asked to choose the most visually promising deblurred image. As shown in Figure 5(b), our proposed method exhibits robust performance on real-world locally blurred images, with sharper reconstructed edges and consistent content. Moreover, it is the most preferred method among the participants when compared to other approaches.

4.3 Analyses

To further analyze the ability of our proposed method, we analyze the effectiveness of the window pruning strategy and blur mask annotations in this section, respectively. Due to the limited space, the analyses of feature channels are put in the supplementary material.

4.3.1 Window pruning strategy

We analyze the effects of our window pruning strategy in two aspects: the number of pruning blocks and the pruning threshold β during inference.

The number of pruning blocks. We fix the pruning threshold $\beta = 0.5$ and change the number of pruning blocks. For blocks without pruning, we force the decision maps \mathbf{D} in Equation (4) to be the all-one matrix. From line 1 to line 3 of Table 4, we find that pruning more blocks results in fewer model parameters, a higher inference speed, and more dropping scores. Notably, although we apply a window pruning mechanism in all 9 blocks, the scores outperform other baseline models listed in Table 1. Additionally, we conduct visual comparisons of all-pruned architecture (line 1) with non-pruning architecture (line 3), as shown in Figure 7. In the all-pruned network, most of the sharp regions are pruned and are not destroyed. In contrast, the sharp regions turn blurry or distorted when processed by the non-pruned network. This indicates that our adaptive window pruning mechanism can prevent the sharp regions from being destroyed. The non-pruning network treats every region equally like other global deblurring networks (e.g., Uformer [24], Restormer[28]) that may harm the sharp regions inadvertently when deblurring. We provide results of pruning 3~7 blocks and 4~6 blocks in the supplementary material.

Table 2: The effectiveness of window pruning. “PSNR_w”, “SSIM_w”, “Time”, and “Params” denote weighted PSNR, weighted SSIM, inference time, and model parameters, respectively.

No.	Pruning location	β	\uparrow PSNR	\uparrow SSIM	\uparrow PSNR _w	\uparrow SSIM _w	Time	Params	FLOPs
1	Block 1~9	0.5	35.43	0.9288	30.29	0.8934	0.56s	54.50M	1.592T
2	Block 2~8	0.5	35.41	0.9290	30.34	0.8928	0.70s	53.94M	1.485T
3	None	0.5	35.36	0.9281	30.24	0.8935	1.30s	50.39M	4.376T
4	Block 1~9	0.2	35.37	0.9289	30.21	0.8931	0.95s	54.50M	1.911T
5	Block 1~9	0.3	35.42	0.9291	30.28	0.8934	0.80s	54.50M	1.671T
6	Block 1~9	0.4	35.44	0.9290	30.30	0.8935	0.69s	54.50M	1.556T
7	Block 1~9	0.6	35.35	0.9284	30.17	0.8921	0.49s	54.50M	1.405T
8	Block 1~9	0.7	35.32	0.9281	30.14	0.8918	0.43s	54.50M	1.327T

Table 3: The effectiveness of our blur mask annotation. “PSNR_w” and “SSIM_w” denote weighted PSNR, and weighted SSIM, respectively.

No.	Methods	\uparrow PSNR	\uparrow SSIM	\uparrow PSNR _w	\uparrow SSIM _w	\uparrow Accuracy	\uparrow Recall	\uparrow Precision
1	LBAG (with LBFMG)[10]	34.83	0.9264	28.31	0.8711	0.8978	0.9089	0.6632
2	LBAG [10] (with ours)	34.90	0.9262	29.60	0.8866	0.9173	0.9209	0.6949
3	LMD-ViT (with LBFMG [10])	35.30	0.9272	30.20	0.8920	/	/	/
4	LMD-ViT (with ours)	35.43	0.9288	30.29	0.8934	/	/	/

Pruning threshold β . We fix the pruning blocks and adjust the pruning threshold β from 0.2 to 0.7 with 0.1 as the interval. Comparing line 1, lines 4 to 8 in Table 4, we find that the testing performance slightly varies with different pruning thresholds β , resulting in different confidence levels and decision boundaries. Inferring with a lower (e.g., $\beta = 0.2$) or a higher (e.g., $\beta = 0.7$) pruning threshold is neither reasonable, because the former makes the confidence predictor more inclusive, which potentially leads to fewer sharp windows to be pruned and a slower inference speed, while the latter makes the confidence predictor more conservative, which leads to faster inference speed but filters out the necessary blurry windows. To achieve a balance, we choose $\beta = 0.5$ as it obtains relatively high evaluation scores and fast inference speed.

4.3.2 Blur mask annotation

We conduct several experiments to verify the effectiveness of our blur mask annotations on various local motion deblurring networks, as illustrated in Table 3. For LBAG [10] which predicts blur masks as gates for local deblurring, our manually annotated blur masks enhance both image similarity and blur detection accuracy, comparing line 1 and line 2. Figure 6 also shows that the LBAG blur detection restrained by LBFMG masks [10] leads to more blur detection errors. When testing on LMD-ViT, our blur mask annotations improve the evaluation metrics obviously. For fair comparisons, we cannot evaluate the detection accuracy/recall/precision because LMD-ViT does not explicitly learn bur detection. The binary masks generated by LBFMG [10] contain holes in blurred regions and noise in sharp regions, which may confuse AdaWTP to select helpful tokens.

5 Conclusion

In this paper, we presented an adaptive and efficient approach, LMD-ViT, for restoring high-resolution images affected by local motion blurs. LMD-ViT is built upon our novel adaptive window pruning Transformer blocks (AdaWPT), which utilize blur-aware confidence predictors to estimate the level of blur confidence in the feature domain. This information is then used to adaptively prune unnecessary windows in low-confidence regions. To train the confidence predictor, we designed an end-to-end reconstruction loss with Gumbel-Softmax re-parameterization, along with a pruning loss guided by our meticulously annotated blur masks. Extensive experiments demonstrate that our method effectively eliminates local motion blur while ensuring minimal deformation of sharp regions, resulting in a significant improvement in image quality and inference speed. Due to limited page space, we discuss the limitations and broader impacts in the supplementary material.

References

- [1] Daniel Bolya, Cheng-Yang Fu, Xiaoliang Dai, Peizhao Zhang, Christoph Feichtenhofer, and Judy Hoffman. Token Merging: Your vit but faster. In *ICLR*, 2022.
- [2] Liangyu Chen, Xin Lu, Jie Zhang, Xiaojie Chu, and Chengpeng Chen. HINet: Half instance normalization network for image restoration. In *CVPR*, 2021.
- [3] 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.
- [4] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. In *ICLR*, 2021.
- [5] Lanqing Guo, Siyu Huang, Ding Liu, Hao Cheng, and Bihan Wen. Shadowformer: Global context helps image shadow removal. In *AAAI*, 2023.
- [6] Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with gumbel-softmax. In *ICLR*, 2017.
- [7] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *ICLR*, 2015.
- [8] Orest Kupyn, Volodymyr Budzan, Mykola Mykhailych, Dmytro Mishkin, and Jiří Matas. Deblurgan: Blind motion deblurring using conditional adversarial networks. In *CVPR*, 2018.
- [9] Orest Kupyn, Tetiana Martyniuk, Junru Wu, and Zhangyang Wang. DeblurGAN-v2: Deblurring (orders-of-magnitude) faster and better. In *ICCV*, 2019.
- [10] Haoying Li, Ziran Zhang, Tingting Jiang, Peng Luo, and Huajun Feng. Real-world deep local motion deblurring. In *AAAI*, 2023.
- [11] Jingyun Liang, Jiezhong Cao, Guolei Sun, Kai Zhang, Luc Van Gool, and Radu Timofte. SwinIR: Image restoration using swin transformer. In *ICCVW*, 2021.
- [12] Youwei Liang, Chongjian Ge, Zhan Tong, Yibing Song, Jue Wang, and Pengtao Xie. Not all patches are what you need: Expediting vision transformers via token reorganizations. In *ICLR*, 2022.
- [13] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin Transformer: Hierarchical vision transformer using shifted windows. In *ICCV*, 2021.
- [14] Ilya Loshchilov and Frank Hutter. Sgdr: Stochastic gradient descent with warm restarts. In *ICLR*, 2017.
- [15] Lingchen Meng, Hengduo Li, Bor-Chun Chen, Shiyi Lan, Zuxuan Wu, Yu-Gang Jiang, and Ser-Nam Lim. AdaViT: Adaptive vision transformers for efficient image recognition. In *CVPR*, 2022.
- [16] Seungjun Nah, Tae Hyun Kim, and Kyoung Mu Lee. Deep multi-scale convolutional neural network for dynamic scene deblurring. In *CVPR*, 2017.
- [17] Yongming Rao, Wenliang Zhao, Benlin Liu, Jiwen Lu, Jie Zhou, and Cho-Jui Hsieh. Dynamicvit: Efficient vision transformers with dynamic token sparsification. In *NeurIPS*, 2021.
- [18] Wenqi Ren, Jiawei Zhang, Jinshan Pan, Sifei Liu, Jimmy Ren, Junping Du, Xiaochun Cao, and Ming-Hsuan Yang. Deblurring dynamic scenes via spatially varying recurrent neural networks. *TPAMI*, 2021.
- [19] Kevin Schelten and Stefan Roth. Localized image blur removal through non-parametric kernel estimation. In *ICCV*, 2014.
- [20] Xin Tao, Hongyun Gao, Xiaoyong Shen, Jue Wang, and Jiaya Jia. Scale-recurrent network for deep image deblurring. In *CVPR*, 2018.

- [21] Ashish Vaswani, Prajit Ramachandran, Aravind Srinivas, Niki Parmar, Blake Hechtman, and Jonathon Shlens. Scaling local self-attention for parameter efficient visual backbones. In *CVPR*, 2021.
- [22] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NeurIPS*, 2017.
- [23] Wenhai Wang, Enze Xie, Xiang Li, Deng-Ping Fan, Kaitao Song, Ding Liang, Tong Lu, Ping Luo, and Ling Shao. Pyramid vision transformer: A versatile backbone for dense prediction without convolutions. In *ICCV*, 2021.
- [24] 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.
- [25] Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing*, 13(4):600–612, 2004.
- [26] Jianwei Yang, Chunyuan Li, Pengchuan Zhang, Xiyang Dai, Bin Xiao, Lu Yuan, and Jianfeng Gao. Focal self-attention for local-global interactions in vision transformers. In *NeurIPS*, 2021.
- [27] Hongxu Yin, Arash Vahdat, Jose M Alvarez, Arun Mallya, Jan Kautz, and Pavlo Molchanov. A-ViT: Adaptive tokens for efficient vision transformer. In *CVPR*, 2022.
- [28] 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.
- [29] 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.
- [30] Jiawei Zhang, Jinshan Pan, Daoye Wang, Shangchen Zhou, Xing Wei, Furong Zhao, Jianbo Liu, and Jimmy Ren. Deep dynamic scene deblurring from optical flow. *IEEE Transactions on Circuits and Systems for Video Technology*, 32(12):8250–8260, 2021.
- [31] Shanghang Zhang, Xiaohui Shen, Zhe Lin, Radomír Měch, Joao P Costeira, and José MF Moura. Learning to understand image blur. In *CVPR*, 2018.
- [32] Shangchen Zhou, Chongyi Li, and Chen Change Loy. LEDNet: Joint low-light enhancement and deblurring in the dark. In *ECCV*, 2022.

Adaptive Window Pruning for Effective Local Motion Deblurring

– Supplementary Material –

This supplementary document is organized as follows:

- Section A provides a detailed description of our proposed Transformer-based local motion deblurring architecture, LMD-ViT.
- Section B further discusses the effectiveness of window pruning, the window-based locally-enhanced feed-forward layer (W-LeFF), and the effect of the numbers of feature channels, respectively.
- Section C provides more visual results between our proposed LMD-ViT and other baseline methods [16, 9, 2, 3, 10, 28, 24], including local motion deblurring results on the ReLoBlur dataset [10], global deblurring performance on the GOPRO dataset [16], and user-study results of real-world local motion deblurring.
- Section D explains our method’s limitations and broader impacts.
- Additionally, we provide the link to our code and blur mask annotation ³.

A LMD-ViT Architecture and Model Hyper-parameters

As shown in Figure 2 of our main paper, our proposed LMD-ViT is a U-shaped network with one in-projection layer, four encoder stages, one bottleneck stage, four decoder stages, and one out-projection layer. Skip connections are set up between the encoder stage and the decoder stage. A locally blurred input image $\mathbf{B} \in \mathbb{R}$ with a shape $H \times W \times 3$ firstly goes through an in-projection block, which consists of a 3×3 convolutional layer, a LeakyReLU layer, and a layer normalization block, to extract low-level features as a feature map $\mathbf{X} \in \mathbb{R}$. The feature map then passes four encoder stages, each of which includes a series of AdaWPT Transformer blocks and one down-sampling layer. AdaWPT uses a blur-aware confidence predictor and Gumble-Softmax re-parameterization to select blur-related tokens. Only the selected tokens are forwarded to Transformer layers including window-based self-attention (W-MSA), window-based locally-enhanced feed-forward layer (W-LeFF), and layer normalization (LN). The down-sampling layer down-samples the feature map size by 2 times and doubles the channels using 4×4 convolution with stride 2. The feature map’s shape turns to $\frac{H}{2^i} \times \frac{W}{2^i} \times 3, i \in \{1, 2, 3, 4\}$ after i encoder stages, and has the smallest resolution in the bottleneck, where it can sense the longest dependencies in two AdaWPTs. After the bottleneck stage, the feature map goes through four decoder stages, each of which owns an up-Sampling layer and a series of AdaWTP blocks. The up-sampling layer uses 2×2 transposed convolution with stride 2 to reduce half of the feature channels and double the size of the feature map. The features put into the AdaWPT blocks are concatenations of the up-sampled features and the corresponding features from the symmetrical encoder stages through skip connections. Finally, the feature map passes the out-projection block which reshapes the flattened features to 2D feature maps and applies a 3×3 convolution layer to obtain a residual image \mathbf{R} . The restored sharp image \mathbf{S}' is obtained by $\mathbf{S}' = \mathbf{B} + \mathbf{R}$.

³<https://drive.google.com/drive/folders/1pX1Tbjz3NaZcOaFFRZkIomotupFMz0lr?usp=sharing>

Table 4: The effectiveness of window pruning. “PSNR_w”, “SSIM_w”, “Time”, “Params”, and “FLOPs” denote weighted PSNR, weighted SSIM, inference time, model parameters, and model complexity respectively.

No.	Pruning block	β	\uparrow PSNR	\uparrow SSIM	\uparrow PSNR _w	\uparrow SSIM _w	Time	Params	FLOPs
1	Block 1~9	0.5	35.43	0.9288	30.29	0.8934	0.56s	54.50M	1.485T
2	Block 2~8	0.5	35.41	0.9290	30.34	0.8928	0.70s	53.94M	1.592T
3	Block 3~7	0.5	35.45	0.9290	30.40	0.8937	0.77s	53.49M	2.021T
4	Block 4~6	0.5	35.36	0.9293	30.39	0.8931	1.07s	53.06M	3.105T
5	None	/	35.36	0.9281	30.24	0.8935	1.30s	50.39M	4.376T

B Further Discussion

B.1 Discussions of window pruning

In this section, we first discuss the influence of the number of pruning blocks by presenting more quantitative results. Then we analyze the pruning granularity and pruning precision by showing visualization results. To explain specifically, we number the four encoder stages from 0 to 4, the middle stage as 5, and the decoder stage from 6 to 9.

The number of pruning blocks We set the pruning threshold to $\beta = 0.5$ and vary the number of pruning blocks. The results in Table 4 show that pruning more blocks leads to an increase in model parameters, as more confidence predictors and decision layers are inserted. However, increasing the number of pruned blocks reduces inference time and FLOPs by removing tokens from processing. It is worth noting that the all-pruned network achieves a comparable score with the non-pruned network, and outperforms other baseline models listed in Table 2 of the main paper. To achieve a balance between computational efficiency and evaluation scores, we select the all-pruned network as our backbone, where blocks 1 to 9 are pruned. This pruning strategy offers the shortest inference time while maintaining comparable evaluation scores compared to other pruning strategies listed in Table 4.

Pruning precision We provide more examples of blur mask annotation and visualizations of window pruning in different AdaWPT blocks in Figure 8. The visual results reveal two important observations. Firstly, our proposed LMD-ViT algorithm prunes windows more coarsely in low-resolution blocks and more finely in high-resolution blocks. This approach strikes a balance between computational complexity and accuracy. Secondly, our proposed adaptive window pruning Transformer block (AdaWPT) enables blur region selection as well as removing blur-irrelevant windows. From block 4 to block 8, the preserved patches gradually correspond to the ground-truth blur masks. To verify whether blurred regions are mistakenly pruned, we calculate the precision of window pruning by:

$$Precision = \frac{TP}{TP + FP},$$

where TP refers to the number of pixels which our model correctly predicts to be blurry and preserved to process. FP refers to the number of pixels which our model incorrectly identifies to be blurry but are actually sharp, which are also pixels that our model accidentally preserved but should be actually removed. The average precision per block in the ReLoBlur testing dataset [10] is 96.8%, suggesting that the preserved windows cover most blur regions and AdaWPT significantly avoids pruning blurry regions.

B.2 The effectiveness of W-LeFF

To save computational costs as well as inference time, we apply a window-based local-enhanced feed-forward layer (W-LeFF) instead of a locally-enhanced feed-forward layer (LeFF) [24]. In W-LeFF, only the non-pruned windows are processed by a feed-forward mechanism. To ensure a fair comparison between W-LeFF and LeFF, we evaluate them on the all-pruning LMD-ViT network architecture separately. As shown in Table 5, we observe that W-LeFF achieves nearly identical performance to LeFF. Hence, substituting LeFF with W-LeFF does not compromise the local motion deblurring capabilities while simultaneously accelerating the inference speed.

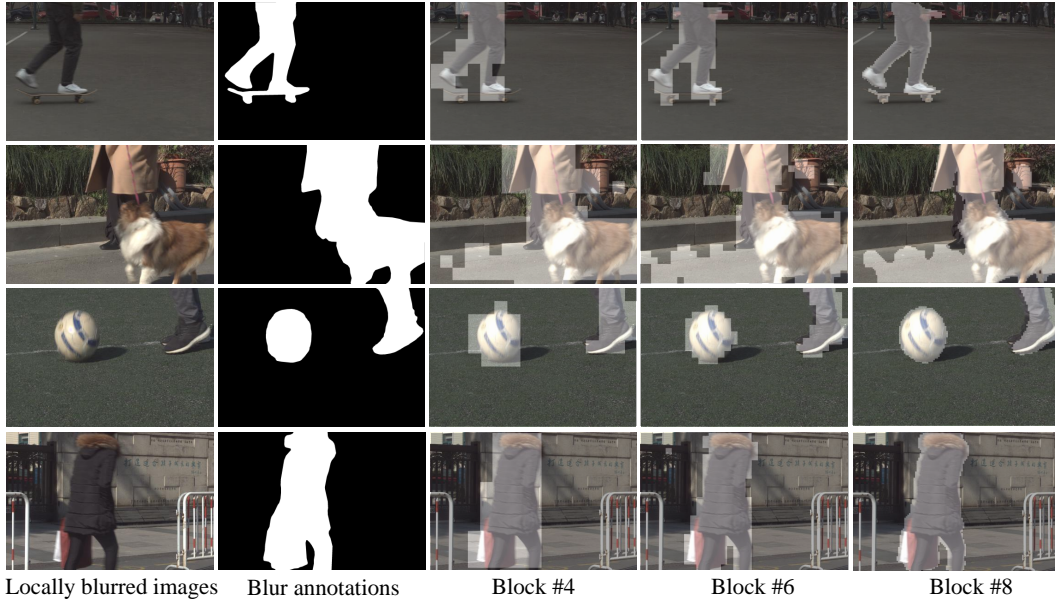


Figure 8: Visualizations of window pruning mechanism in different blocks of our network. The white color denotes the preserved patches, which cover nearly all blurry regions. From block 4 to block 8, the preserved patches gradually correspond to the ground-truth blur masks and AdaWPT prunes more finely.

Table 5: The effectiveness of W-LeFF in the proposed LMD-ViT. “PSNR_w”, “SSIM_w”, “Time”, “Params”, and “FLOPs” denote weighted PSNR, weighted SSIM, inference time, model parameters, and model complexity, respectively.

No.	Methods	Pruning block	↑PSNR	↑SSIM	↑PSNR _w	↑SSIM _w	Time	Params	FLOPs
1	LMD-ViT w LeFF	Block 1~9	35.45	0.9288	30.24	0.8935	0.86s	54.50M	1.485T
2	LMD-ViT w W-LeFF	Block 1~9	35.43	0.9288	30.29	0.8934	0.56s	54.50M	1.485T

B.3 The effect of feature channels

The number of feature channels also affects the ability of neural networks. With a larger number of feature channels, a neural network can capture more intricate and complex relationships in the input data, resulting in better performance. To verify the capability of our proposed network architecture, we train our network with dimensions 16 and 32 respectively, and compare it with CNN-based LBAG [10] with aligned model parameters, as shown in Table 6. The comparison between line 3 and line 4 shows an improvement with increased feature channels in LMD-ViT because a larger number of feature channels provides a larger number of meaningful features or attributes, which can be beneficial for window selection and feature manipulation. The comparison between line 2 and line 4 implies that, under approximate model parameters, our proposed model with the adaptive window pruning mechanism is more suitable for the local motion deblurring task with better evaluation scores, fewer model parameters, and faster inference speed.

Table 6: The effect of feature dimension in the proposed LMD-ViT. “Feature channels”, “PSNR_w”, “SSIM_w”, “Time”, “Params”, and “FLOPs” denote the feature channels of each block, weighted PSNR, weighted SSIM, inference time, model parameters, and model complexity respectively.

No.	Methods	Feature channels	↑PSNR	↑SSIM	↑PSNR _w	↑SSIM _w	Time	Params	FLOPs
1	LBAG [10]	32-64-128	34.90	0.9262	29.60	0.8866	0.51s	16.11M	7.852T
2	LBAG-Large [10]	60-120-240	34.93	0.9266	29.63	0.8871	1.13s	56.58M	17.659T
3	LMD-ViT-Small	16-32-64-128-256	34.98	0.9259	29.89	0.8907	0.23s	21.59M	0.311T
4	LMD-ViT	32-64-128-256-512	35.43	0.9288	30.29	0.8934	0.56s	54.50M	1.592T

C More Visual Results

C.1 More visual comparisons between LMD-ViT and the compared methods

We provide more visual results of our proposed LMD-ViT and other baseline methods for local motion deblurring, as shown in Figure 9 and Figure 10. The performance of LMD-ViT surpasses that of other state-of-the-art methods, yielding output images with improved clarity and enhanced details.

C.2 More visual results of LMD-ViT for the global deblurring task

To prove that our proposed LMD-ViT could also deblur globally, we test our model on the GOPRO testing dataset [16]. With a globally blurry input, the predicted blurriness confidence approaches 100%. Therefore, LMD-ViT preserves all windows, represented by the white color in the 5th column of Figure 11. As shown in Figure 11, LMD-ViT effectively eliminates global blurriness and restores sharp textures.

C.3 More visual results of real-world local motion deblurring

We provide more visual results for real-world local motion deblurring, which are used for conducting user study, as shown in Figure 12. We conduct a comparison between our method and the top 3 methods listed in Table 1 of the main paper. The visual results show that our proposed method exhibits robust deblurring capability and outperforms the state-of-the-art methods on real-world locally blurred images.

D Limitations and Broader Impacts

Limitations Like other local motion deblurring methods [16, 8, 9, 2, 3, 10, 24, 28], our proposed LMD-ViT is not capable of real-time deblurring. However, we plan to optimize our structure so that it can be applied in real time.

Broader impacts Our proposed local motion deblurring method democratizes access and empowers users across multiple fields to enhance their visual content, fostering creativity and expression. Firstly, in the realm of consumer electronics and mobile devices, where camera stabilization technology like gimbals has mitigated global motion blur caused by camera shake, local motion blur remains a key problem for static cameras. By incorporating effective local motion deblurring algorithms into these devices, we adaptively enhance image quality with less inference time, leading to superior user experiences and heightened customer satisfaction. In the realm of forensic analysis and surveillance, where local motion blur often plagues static surveillance camera images or video frames, our method enhances the quality of these blurred visuals, enabling better object, individual, and event identification. It proves invaluable in criminal investigations, accident reconstructions, and security-related applications. Last but not least, photographers and visual artists benefit greatly, as it salvages blurred images caused by subject motion, resulting in sharper, visually appealing photographs with enhanced details and clarity. This advancement elevates the overall quality of visual information acquisition and visual arts as expressive mediums.

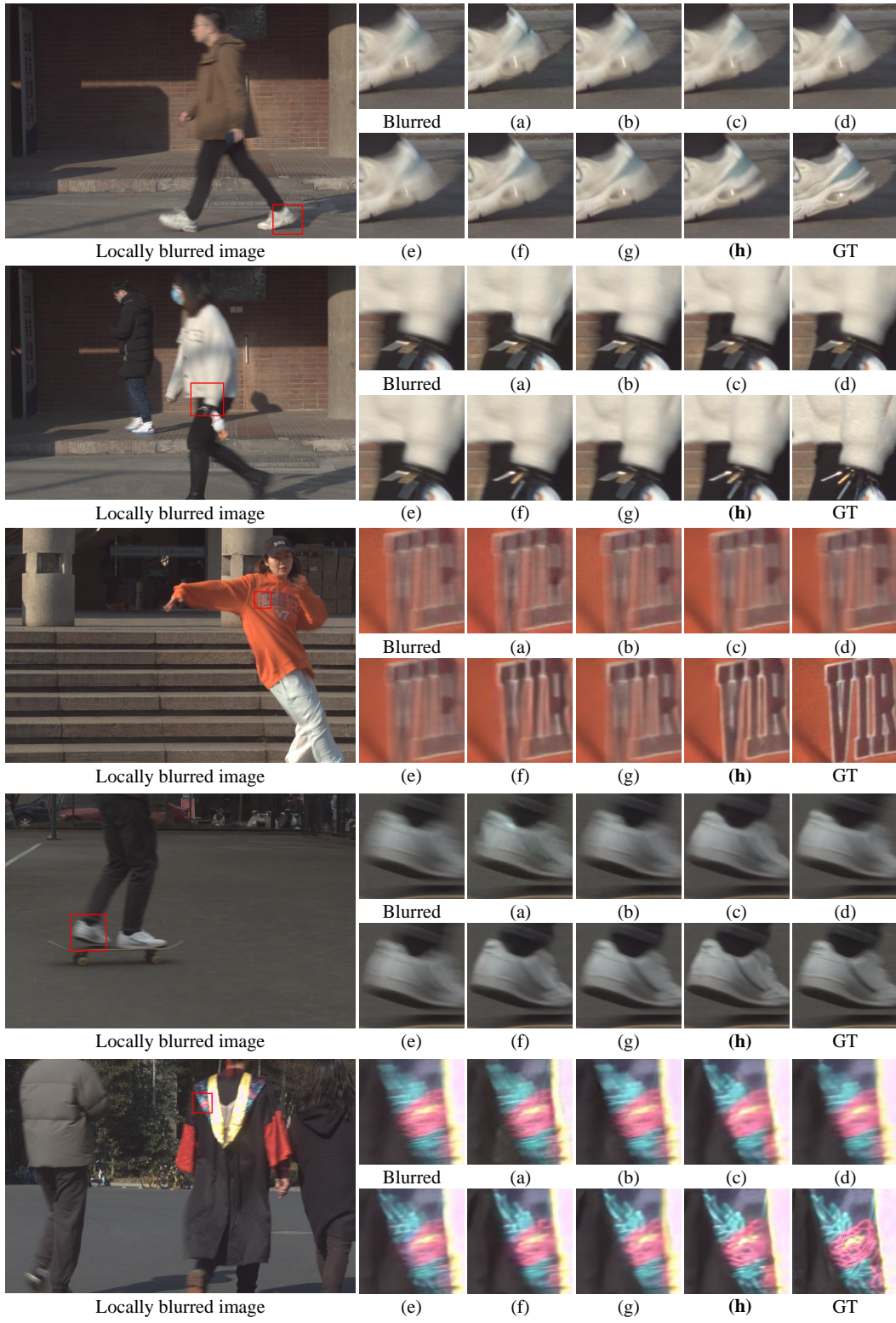


Figure 9: Visual comparisons of state-of-the-art methods for local motion deblurring. (a) DeepDeblur [16]; (b) DeblurGAN_v2 [9]; (c) HINet [2]; (d) MIMO-UNet [3]; (e) LBAG [10]; (f) Restormer [28]; (g) Uformer [24]; (h) LMD-ViT (ours).

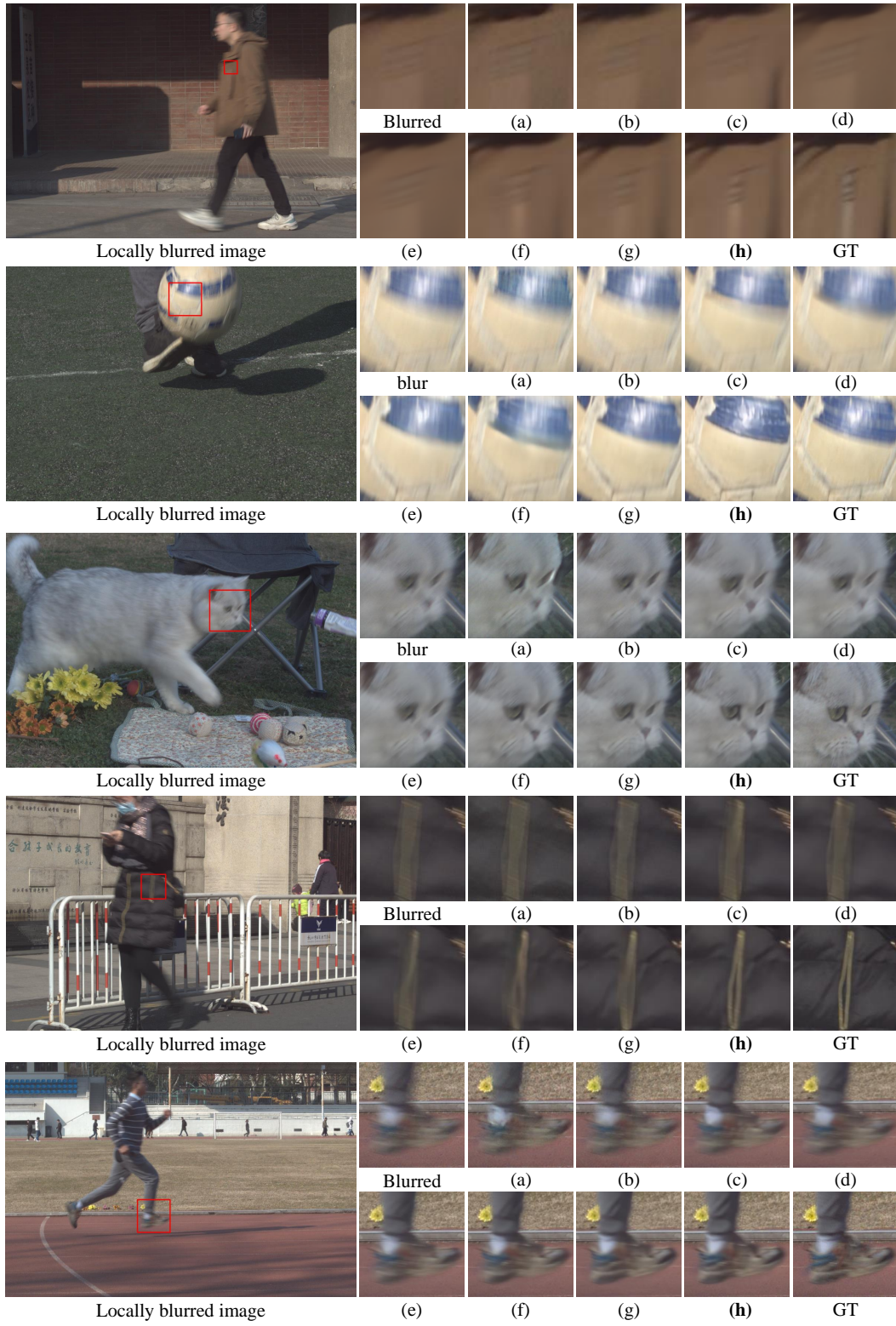


Figure 10: Visual comparisons 2 for local motion deblurring. (a) DeepDeblur [16]; (b) DeblurGAN_v2 [9]; (c) HINet [2]; (d) MIMO-UNet [3]; (e) LBAG [10]; (f) Restormer [28]; (g) Uformer [24]; (h) LMD-ViT (ours).

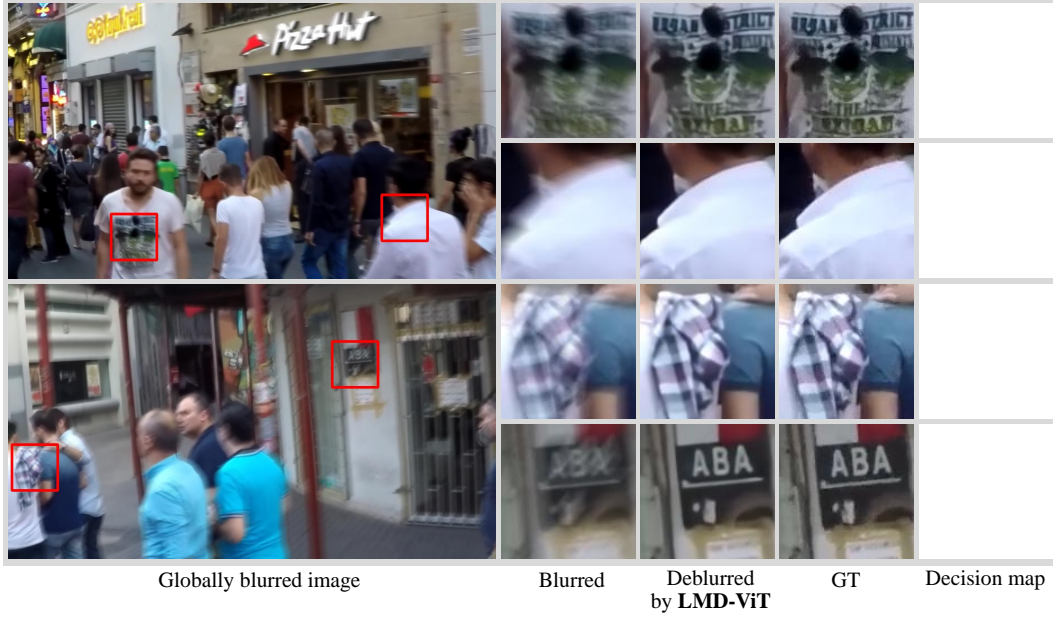


Figure 11: Global deblurring performance of our proposed LMD-ViT. The decision map denotes the pruning decisions. With a globally blurry input, the predicted blurriness confidence approaches 100%. Therefore, our network preserves all windows, which are represented as all-white color.

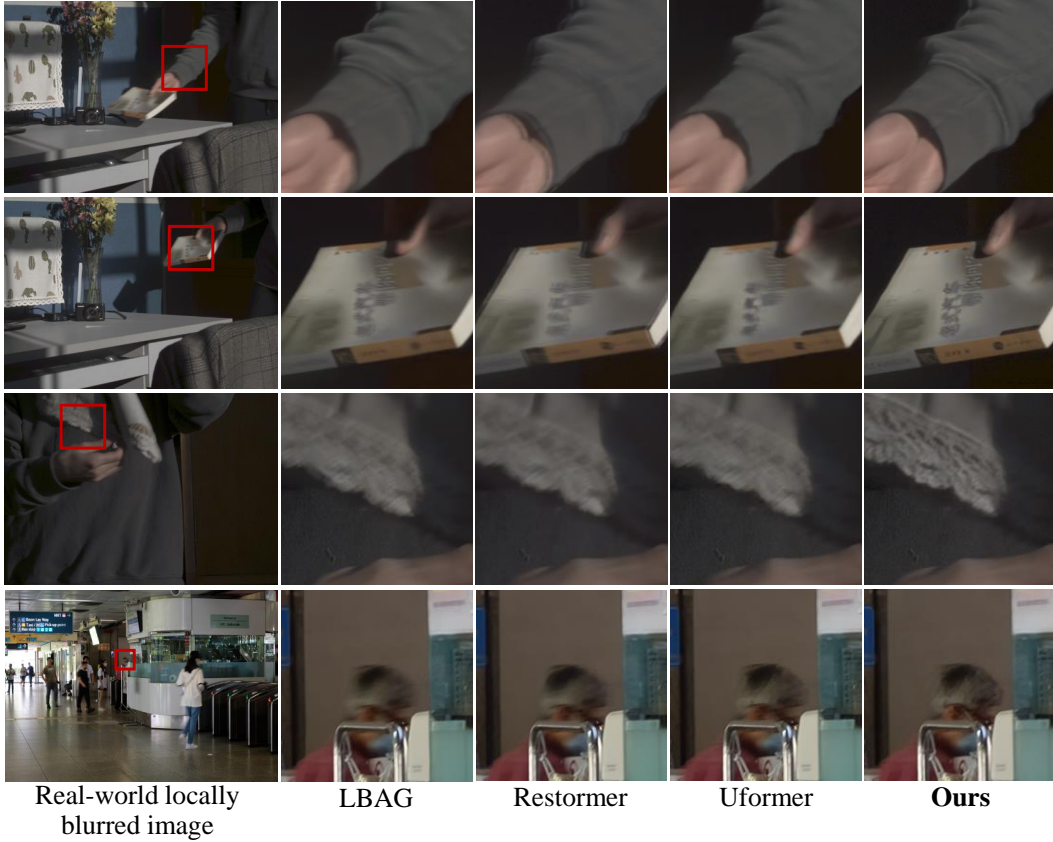


Figure 12: Visual results of user study for real-world local motion deblurring. We compare our proposed LMD-ViT with the top 3 methods, i.e., LBAG[10], Restormer [28], and Uformer [24], listed in Table 1 of the main paper.