International Journal of Computer Vision (2024) 132:1817–1834
https://doi.org/10.1007/s11263-023-01958-9

Fast Ultra High-Definition Video Deblurring via Multi-scale Separable Network

Wenqi Ren[1,2] · Senyou Deng[1] · Kaihao Zhang[3] · Fenglong Song[4] · Xiaochun Cao[1] · Ming-Hsuan Yang[5]

Received: 19 December 2022 / Accepted: 5 November 2023 / Published online: 11 December 2023

Abstract

Despite significant progress has been made in image and video deblurring, much less attention has been paid to process ultra high-definition (UHD) videos (e.g., 4K resolution). In this work, we propose a novel deep model for fast and accurate UHD video deblurring (UHDVD). The proposed UHDVD is achieved by a depth-wise separable-patch architecture, which operates with a multi-scale integration scheme to achieve a large receptive field without adding the number of generic convolutional layers and kernels. Additionally, we adopt the temporal feature attention module to effectively exploit the temporal correlation between video frames to obtain clearer recovered images. We design an asymmetrical encoder–decoder architecture with residual channel-spatial attention blocks to improve accuracy and reduce the depth of the network appropriately. Consequently, the proposed UHDVD achieves real-time performance on 4K videos at 30 fps. To train the proposed model, we build a new dataset comprised of 4K blurry videos and corresponding sharp frames using three different smartphones. Extensive experimental results show that our network performs favorably against the state-of-the-art methods on the proposed 4K dataset and existing 720p and 2K benchmarks in terms of accuracy, speed, and model size.

Keywords Separable-patch · Multi-scale network · Multi-patch network · Ultra high-definition · 4K deblurring

1 Introduction

Ultra High-Definition (UHD, i.e., 12 megapixels or 4K) videos have become a trend during the last several years. Recent consumer electronic devices (e.g., smartphones and DSLR cameras) commonly support 4K videos. Unfortunately, irregular camera shakes and high-speed movements often generate undesirable blurs in captured UHD videos. Low-quality blurred videos make the downstream vision tasks significantly more challenging (Nah et al., 2019b).

Numerous image and video deblurring methods have been proposed to recover the sharp frames from a captured blurry video. Conventional deblurring methods usually make certain assumptions on motion blurs, scene structures, and latent frames to alleviate the nature of the ill-posed problem. Among these methods, motion blurs are usually modeled as uniform kernels (Shan et al., 2008; Zhou & Komodakis, 2014; Zhang et al., 2019) or non-uniform kernels (e.g., region-wise (Ji & Wang, 2012; Bar et al., 2007; Hyun Kim et al., 2013; Wulff & Black, 2014; Cho et al., 2007) and pixel-

Communicated by Chen Change Loy.

✉ Ming-Hsuan Yang
mhyang@ucmerced.edu

Wenqi Ren
renwq3@mail.sysu.edu.cn

Senyou Deng
dengsenyou@gmail.com

Kaihao Zhang
super.khzhang@gmail.com

Fenglong Song
songfenglong@huawei.com

Xiaochun Cao
caoxiaochun@mail.sysu.edu.cn

[1] School of Cyber Science and Technology, Shenzhen Campus, Sun Yat-sen University, 518107 Shenzhen, China

[2] Key Laboratory of Education Informatization for Nationalities, Ministry of Education, Yunnan Normal University, Kunming 650500, China

[3] Department of Computer Science, Australian National University (ANU), Pok Fu Lam, Hong Kong, China

[4] Huawei Noah's Ark Lab, Beijing 100196, China

[5] School of Engineering, University of California at Merced, Merced, CA 95343, USA

wise (Hyun Kim & Mu Lee, 2014; Ren et al., 2017)). While the sharp frames are usually constrained by hand-crafted image priors (Lai et al., 2015; Sun et al., 2013; Michaeli & Irani, 2014; Hu & Yang, 2015) to regularize the solution space, these assumptions do not usually hold for real cases and thereby lead to inaccurate estimations of blur kernels and low-quality of restored images.

In recent years, deep convolutional neural networks (CNNs) based methods have been developed to explicitly learn features from blurry input and regress the blur kernel (Schuler et al., 2015; Gong et al., 2017; Sun et al., 2015) or directly recover the clean image (Nah et al., 2017; Zhang et al., 2018, 2020; Zhou et al., 2019; Zhang et al., 2019; Hu et al., 2021). There are two common strategies, "multi-scale" and "multi-patch", to exploit the deblurring cues at different processing levels and regions, respectively. These algorithms can remove blur effects caused by camera shakes and object motions, and achieve state-of-the-art results on image deblurring tasks. However, there are two main limitations of existing CNN-based methods. First, the computational and memory costs are prohibitively high for certain applications, especially when high-resolution images need to be processed. For example, the recent video deblurring method of CDVD-TSP (Pan et al., 2020) needs about four seconds and one minute to deblur a single frame from HD (720p) and UHD (4K) videos, respectively. Second, the multi-patch networks neglect scale-variant properties of features, which are crucial for respective restoration in each scale (Gao et al., 2019). Therefore, generating detailed textures from blurred images is still a challenging problem.

In addition, it is important to effectively exploit the temporal information between frames for video deblurring. Deep learning-based methods have recently made significant improvements in video deblurring. Several CNN-based methods (Su et al., 2017; Wang et al., 2019) obtain deblurred frames by simply stacking neighboring frames with the current frame as input. RNN-based schemes (Hyun Kim et al., 2017; Nah et al., 2019a; Wieschollek et al., 2017; Zhong et al., 2020) employ recurrent neural network architecture to transfer visual information across frames for inference. As such, these models either entail high computational costs by concatenating neighboring frames or have limited capacity to efficiently transfer the effective information temporally.

In this work, we propose a novel UHDVD network with high efficiency, low memory consumption, and high-quality deblurring performance. Our method is motivated by patch-hierarchical image deblurring methods (Zhang et al., 2019; Suin et al., 2020), where a multi-patch hierarchy is fed into the network. These schemes are able to deblur images of 720p well. However, the multi-patch hierarchy has the same spatial resolution at different levels and requires the layout of patches and stitching, which limits the quality of reconstructed images and reduces the runtime performance. In addition, it is more challenging to process high-resolution or ultra-high-resolution images. Thus, we propose a separable-patch model combined with a multi-scale integration scheme, which captures the global structure and processes multiple patches of each scale simultaneously.

While most existing deblurring algorithms employ cascaded networks to help restore latent frames (Pan et al., 2020; Zhang et al., 2019), simply stacking the same network to construct deeper models may not restore images well (Suin et al., 2020). In this work, we propose a cascaded residual channel and spatial attention (RCSA) module and a Temporal Feature Attention (TFA) module to improve deblurring performance without sacrificing runtime performance. The proposed RCSA is able to adaptively learn useful channel-wise features and emphasize the most informative region on the feature map. At the same time, the TFA module can extract correlation features for neighboring frames.

The main contributions of this work are:

- We propose a novel UHDVD network using a separable-patch architecture combined with a multi-scale integration scheme. The proposed model is the first deep video deblurring model to deblur 4K videos in real-time by parallelizing multiple patches.
- We introduce a Temporal Feature Attention (TFA) module to improve the utilization of the correlation between video frames.
- We design a cascaded RCSA module to improve feature representation power and discriminative ability, ensuring high deblurring performance.
- We construct a 4K deblurring dataset (4KRD) including synthesized and real captured videos. Extensive experimental results on the proposed and existing benchmark datasets (Nah et al., 2019, 2017; Su et al., 2017) show that our model performs favorably against state-of-the-art approaches.

Preliminary results of this work are published in Deng et al. (2021). In this paper, we extend our prior work in several aspects. First, to make full use of temporal feature between frames, we propose a Temporal Feature Attention module in our improved model. With this module, we obtain more abundant features between frames than directly concatenating previous deblurred frames in Deng et al. (2021). Furthermore, we use depth-wise separable convolution to decrease UHDVD model size and FLOPS. Second, in addition to the 4KRD deblurring dataset, we conduct more experiments on a new 2K resolution dataset [Slow-Flow (Janai et al. 2017)] and real test 720p datasets [DVD (Su et al. 2017) and REDS (Nah et al. 2019)]. Third, we analyze the model size and FLOPS of the proposed network and other state-of-the-art methods, our proposed model achieves a 16× faster runtime than the state-of-the-art methods.

2 Related Work

To address the ill-posed nature of the deblurring problem, numerous methods exploit different priors and assumptions of the scenes, including total variation (Perrone & Favaro, 2014), sparse image priors (Liu et al., 2017; Dong et al., 2011), gradient distributions (Krishnan et al., 2011; Chen et al., 2019), patch priors (Michaeli & Irani, 2014; Sun et al., 2013), and l_0-norm regularizers (Xu et al., 2013; Li et al., 2018). One limitation of these prior-based approaches is that the assumptions or priors do not always hold for dynamic scenes containing depth variations and moving objects.

With the advances of deep learning, CNN-based approaches have also been proposed for image deblurring (Zhang et al., 2018a; Jiang et al., 2020; Nan et al., 2020). The main idea of these models is to learn a mapping function between the blurry input and the corresponding sharp image using a CNN. In addition, Generative Adversarial Nets (GANs) have also been exploited for image deblurring (Kupyn et al., 2019, 2018). As these models usually involve large model parameters and entail heavy computational loads, it is not feasible to apply these approaches to real-time deblurring tasks, especially for UHD videos.

Multi-Scale and Multi-Patch Networks. Coarse-to-fine (i.e., multi-scale) models have been widely used in conventional approaches and recent deep models for deblurring. Nah et al. (Nah et al., 2017) propose a multi-scale CNN-based deblurring network, which starts from a coarse scale of the blurred input and then progressively deblurs the input at higher scales until the full-resolution latent image is recovered. In (Tao et al., 2018), Tao et al. introduce a scale-recurrent network by training shared parameters across scales. The approach can preserve image structures and motion information from the previous coarser scales based on the recurrent network. Gao et al. (Gao et al., 2019) improve the multi-scale CNN (Tao et al., 2018) by selectively sharing parameters and modules at each scale. However, these multi-scale networks are usually large and computationally expensive at the inference stage.

To address these issues, a hierarchical multi-patch model (Zhang et al., 2019) is proposed to exploit motion information at different scales by feature aggregation over multiple patches. Suin et al. (Suin et al., 2020) combine the multi-patch hierarchical and a global attention mechanism without using a cascade of convolutional layers. Recently, Zamir et al. (Zamir et al., 2021) use a similar scheme in a multi-stage architecture to obtain better results at the expense of computational loads. While these multi-patch networks deblur 720p images well in real-time, existing approaches are not designed to process full high-definition (FHD, 1920 × 1080 resolution) inputs or UHD videos (e.g., 4K resolution). Figure 1 summarizes the performance of representative deblurring methods on 720p and UHD images.

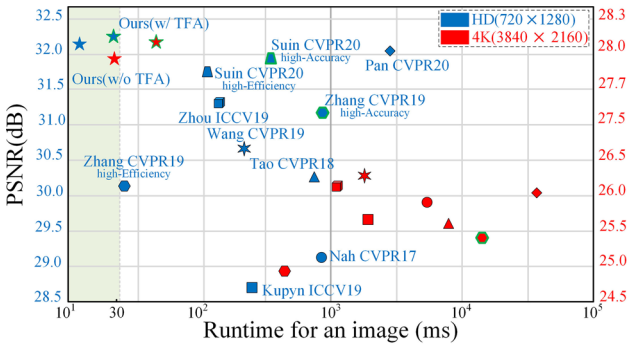

Fig. 1 PSNR (dB) versus runtime (ms) of several deep learning deblurring methods and ours on different datasets. The green region indicates real-time inference less than 30 fps. The blue PSNR and icons are methods on the HD dataset, and the red ones with the same shape are on the 4KRD dataset. The icons with green outlines are the high-accuracy version (w/ TFA). The proposed model performs well not only in efficiency but also in accuracy (Color figure online)

FHD and UHD Image Enhancement. A few methods have been proposed to recover clear images from FHD or UHD degraded inputs by learning bilateral regularizer (Kim et al., 2019) or 3D Lookup Tables (Zeng et al., 2020). However, all these methods reconstruct the final output by sophisticated interpolation techniques from a down-sampled version. In contrast to these approaches, our network directly deblurs images at the full-resolution inputs on the finest scale and is the first real-time deblurring model for 4K videos at 35 fps.

3 Proposed Algorithm

The core idea of the proposed model is to integrate the multi-scale and multi-patch schemes properly, and we introduce a separable-patch strategy to dramatically accelerate reference implementations. The whole architecture of our UHDVD is shown in Fig. 2.

Given a blurred video, the previous deblurred frame D_{i-1} is concatenated with the current blurry frame B_i at the channel dimension as our network input. The concatenated input is then half-downsampled ordinally at four different scales (B^s, $s = 1, 2, 3, 4$), and corresponding sharp images are recovered at each scale. Since temporal information can improve video deblurring results (Zhong et al., 2020), we propose an RNN-based Temporal Feature Attention (TFA) module to extract temporal features at the first scale. We first obtain temporal features t_i and $t^d_{(i-1)}$ of the current blurry frame and the previous deblurred frame simultaneously. The initial temporal state (h_{i-1}/d_{i-1}) of the previous blurry/deblurred frame will be transferred through the whole video clip in the TFA module (see Sec. 3.1). We also add another two temporal features $t^d_{(i-3)}$ and $t^d_{(i-2)}$ of previous deblurred frames to further enhance the temporal relation-

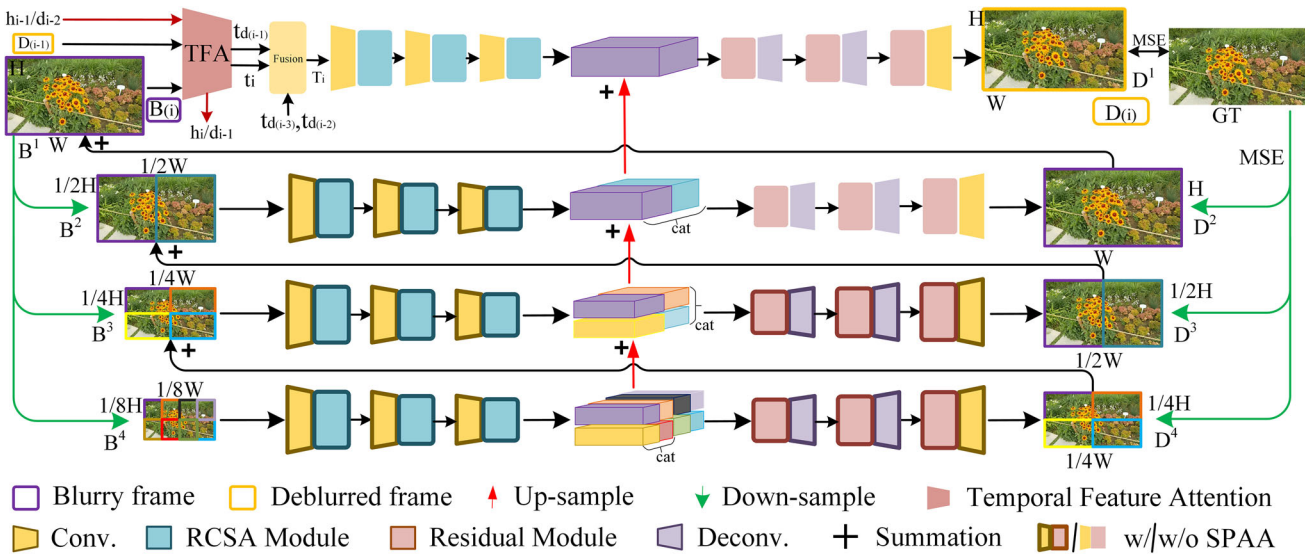

Fig. 2 Architecture of the proposed UHDVD model. The layers and modules with dark color outline are equipped with the support of SPAA module

ship. All these temporal features will be fed into a Fusion module to obtain richer temporal features T_i as the input of the encoder branch.

At other scales, the current blurry frame B_i will be half-downsampled ordinally at three scales (B^2, B^3, B^4), and corresponding sharp images (D^2, D^3, D^4) are recovered at each scale. The above three scales process can obtain more plentiful spatial-guide features F_i^s and deblurred downsample images. The recovered image D^1 at scale 1 is the final output. Based on this scheme, we can set a larger "crop size" in the training process to expand the receptive field, which means more feature information (Chen et al., 2017; Yu & Koltun, 2015; Wang et al., 2018) can be captured and improve the final deblurred results. In addition, the number of split patches in each scale is multiplied by scales. The input for each scale is generated by dividing the original image input (B^s, $s = 1, 2, 3, 4$) into multiple non-overlapping patches. The maximum number of patch for each scale is set as $J = [1,\ 2,\ 4,\ 8]$. The process of scale $s = [2, 3, 4]$ can be formulated as

$$D^s_{i,j}, F^s_{i,j} = Net^s(B^s_{i,j},\ D^{s+1}_{i,j},\ F^{s+1}_{i,j};\ \theta^{p_s}), \tag{1}$$

where i and j denote the video frame index and the patch index, respectively; Net^s represents the proposed 4K video deblurring network at s-th scale with training parameters denoted as θ^{p_s}. Since the network is also recurrent, the middle state features $F^s_{i,j}$ flow across scales from $s+1$ to s.

As shown in Fig. 2, our real-time 4K video deblurring network is composed of 4 similar encoder–decoder architectures at each scale. Each encoder branch contains three convolutions and each convolution layer is followed by a RCSA module. Note that in each decoder branch, the residual modules are in the front of every deconvolutional layer. The red arrows represent the middle feature maps $F^s_{i,j}$ in (1), which is double up-sampled from $F^{s+1}_{i,j}$. The detailed network configurations are shown in Table 1.

3.1 Temporal Feature Attention

Using temporal features between video frames is vital for the video deblurring task to improve the deblurred results. In this paper, we employ a similar RNN-based Temporal Feature Attention module, composed of several Residual Dense Blocks (RDBs). Different from the work in Zhong et al. (2020), our TFA architecture is more straightforward. The main goal is to extract the temporal feature of neighbor frames without the time-consuming spatial encode computation. Aside from the temporal feature between blurry frames, we also compute features between deblurred frames by TFA.

The structure and detailed configurations of TFA are illustrated in Fig. 3. First, the current blurry frame B_i and the previous deblurred frame D_{i-1} will be down-sampled parallelly by a RDB and convolution layers, then concatenated with previous temporal states h_{i-1} and d_{i-2}, respectively. After processing by a series of RDBs and a dense convolution layer, we obtain the intermediate temporal features of the current blurry frame and previous deblurred frame represented as t_i and $t^d_{(i-1)}$. Next, we add another two previously deblurred middle temporal features of $t^d_{(i-3)}$ and $t^d_{(i-2)}$ as inputs of the fusion module. Finally, a dense convolution layer fuses these intermediate temporal features to generate the temporal feature T_i of the current frame. The whole pro-

Table 1 Network configurations

Part layer	Encoder						Decoder					
	conv1/TFA	RCSA1	conv2	RCSA2	conv3	RCSA3	ResB1	deconv1	ResB2	deconv2	ResB3	deconv3
Input	3/3	32	32	64	64	128	128	128	64	64	32	32
Output	32/32	32	64	64	128	128	128	64	64	32	32	3
Kernel	3/–	3	3	3	3	3	3	4	3	4	3	3
Stride	1/–	1	2↓	1	2↓	1	1	2↑	1	4↑↑	1	1
Sum	–/–	–	–	–	–	–	upper scale	RCSA2	–	RCSA1	–	–

Module layer	Residual Block (ResB) -c32/64/128						CAM -c32/64/128				SAM -c32/64/128	
	conv	ReLU	conv	conv	ReLU	conv	avg_pool/max_pool	conv	ReLU	conv	mean/max	conv
Input	32	–	32	32	–	32	32	32	–	4	32	2
Output	32	–	32	32	–	32	32	4	–	32	1	1
Kernel	3	–	3	3	–	3	1	1	–	1	–	3
Stride	1	–	1	1	–	1	–	1	–	1	–	1

The feature maps are downsampled by convolution with stride 2 and upsampled by deconvolution with stride 2/4. The inner components' configurations in network are also shown in bottom part. A RCSA module is composed of two ResBs, a CAM and a SAM

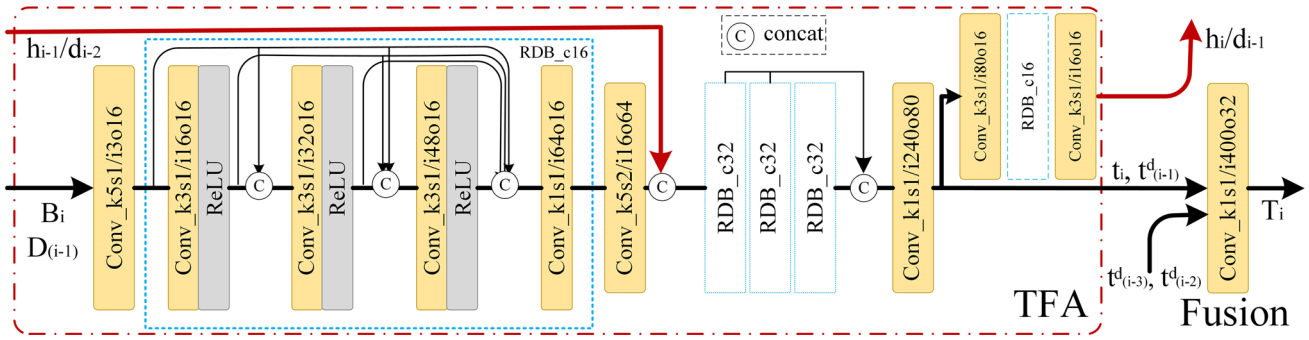

Fig. 3 Proposed temporal feature attention architecture and its configuration. We use h_{i-1}/d_{i-2} and h_i/d_{i-1} to denote the temporal state of blurry/deblurred frames, respectively; t_i and $t^d_{(i-1)}$ refer to temporal features for corresponding blurry and deblurred frames; and T_i refers to the final temporal feature after fusing above four $t_{(...)}$. In each layer, "k#s#/i#o#" denote kernel size, stride, input channel, and output channel. Note that each RDB has the same number of input and output channels

cess can be formulated as

$$T_i = \overset{Fusion}{t_i, t^d_{(...)}}\big(\mathrm{TFA}(B_i/D_{i-1},\ h_{i-1}/d_{i-2};\ \theta)\big), \tag{2}$$

where θ denotes the training parameters, $t^d_{(...)}$ is the intermediate temporal features $t^d_{(i-3,i-2,i-1)}$. Finally, h_i will be updated by t_i through the temporal state generation function which is composed of a RDB and convolution layers.

With the temporal feature T_i extracted from TFA and fusion modules, the process of scale 1 can be formulated as

$$D^1_i = Net^1(T_i,\ F^2_i;\ \theta^{p_1}), \tag{3}$$

where θ^{p_1} denotes the network parameters. F^2_i refers to the middle spatial feature of scale 2.

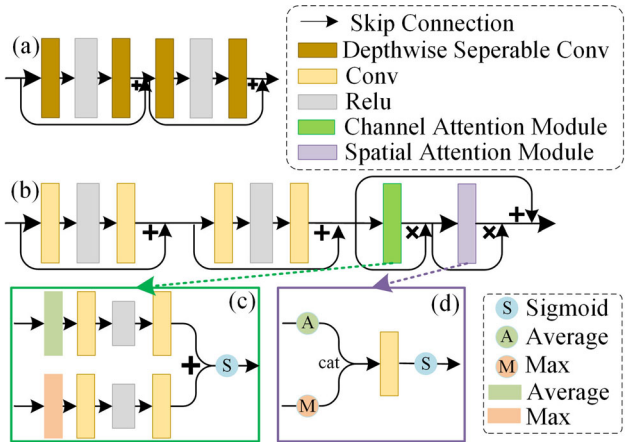

Fig. 4 **a** Structure of residual module with Depth-wise Separable Convolution (DSC). **b** Proposed RCSA module in UHDVD. **c** and **d** are CAM and SAM in RCSA module. Symbol "×" is point-wise multiplication and "+" is addition

3.2 Asymmetrical Encoder–Decoder Architecture

The symmetrical encoder–decoder structure is widely used for visual tasks due to its effectiveness in expanding the receptive field. When dealing with the challenge of increasing feature regions and computing demands caused by 4K resolution, we propose a novel asymmetrical encoder–decoder structure based on the super-resolution framework (Tao et al., 2017) in our architecture, which aims to reduce computation while maintaining performance.

In our architecture, asymmetry is mainly achieved through the differential selection of modules in the encoder and decoder branches. Specifically, on the decoder branch, we employ three lightweight residual modules (Fig. 4a) before each standard deconvolution, effectively reducing the parameters and FLOPS. Each of the three residual modules comprises two Depthwise Separable Convolutions (DSC) with a ReLU activation function in between. This optimization significantly improves computation speed without compromising quality. On the encoder branch, in contrast, we utilize normal convolution layers within the RCSA module (Fig. 4b) instead of depthwise separable convolutions on the decoder branch. We also introduce the temporal-spatial attention mechanism on the encoder branch to further enhance the capabilities of our model. Moreover, to adapt the specific requirements of the encoder and decoder branches, the channel dimensions of the convolution and deconvolution operations are adjusted asymmetrically.

3.3 Separable-Patch Acceleration Architecture

To furtherly improve the inference speed of the UHDVD model, we design the separable-patch acceleration architecture (SPAA) to handle multiple patches or feature maps at the same time. As shown in Fig. 5, the multiple patches (e.g., $n = 4$) are concatenated together as a new tensor in channel dimension at the beginning, and its size is

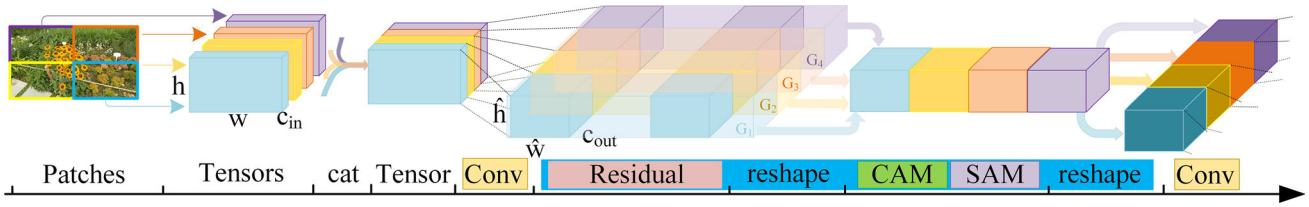

Fig. 5 Separable-patch acceleration architecture. Taking the encoder branch of scale 3 before the second RCSA module as an example, where G_i is the group number in convolutional layers

$[batch\_size,\ n{*}c_{in},\ h,\ w]$. Then, the tensor is processed by the subsequent convolutional layer by setting the parameter $groups = n$. Although this transformation process is almost equivalent to concatenating patches in the batch dimension, these two operations have a subtle difference. When concatenating separable patches in the batch dimension, this augmentation of the effective batch size can lead to potential influences with loss convergence during training and may result in the network converging to local optima less effectively (Keskar et al., 2016).

The computing load of the new tensor is $((n * c_{in}) * (n * c_{out}) * kernel\_size^2)/groups$, while it is equal to n original tensors. But the benefits are that we can change these n serial computations to parallel computation, significantly reducing the computation time. After the computation in residual modules, we reshape the tensor to the size $[batch\_size * n,\ c_{out},\ h,\ w]$ so that it can be computed synchronously in the channel attention module and spatial attention module, respectively. The output will be taken as the input of the next RCSA module and this acceleration is going to continue until we obtain the middle feature maps or restored images of the scale. The SPAA location in Fig. 2 is depicted on layers and modules with a dark outline in scales of $s = [2, 3, 4]$. When computing temporal features of blurry and deblurred frames in the TFA module concurrently, we use similar measures to speed up. By the acceleration of this architecture, our processing speed is twice as fast as the original version without this module. Note that our SPAA strategy is only applied at 1/8, 1/4, and the first half of 1/2 scales processing. This is because processing separable patches at full resolution may lead to artifacts at the patch boundaries.

3.4 Residual Channel-Spatial Attention

We propose a new RCSA module (blue blocks in Fig. 2) that contains a channel attention module and a spatial attention module (Zhao et al., 2020; Gao et al., 2019) in the deblurring network. The architecture of the RCSA is shown in Fig. 4b. Following the recent success of transformer architecture in image processing tasks (Suin et al., 2020; Parmar et al., 2018), the main building blocks of RCSA are channel attention and spatial attention, which calculates the response at channel and spatial dimensions.

The Channel Attention Module (CAM) shown in Fig. 4c comprises two adaptive pooling computations: average pooling and maximum pooling. A standard convolutional layer follows each pooling layer. The output channel is 1/8 of the input channel, kernel size is 1×1 and `bias` is false. Then, there is a ReLU activation function and another same convolution in which input and output channels are exactly opposite of the front convolution. Finally, two processed pooling results are added together as the input of the $sigmoid$ function.

The Spatial Attention Module (SAM) only has one convolutional layer with input channel 2, output channel 1, kernel size is 3×3, padding size is 1, and bias is false. The input data is firstly processed by average and maximum calculation, respectively, and then concatenated together at the channel dimension.

The output of the RCSA module O_{RCSA} is computed by

$$O_{RCSA} = S(O_{CAM}) \times O_{CAM} + I_{CAM}, \tag{4}$$

where I_{CAM} and O_{CAM} are the input and output features of the CAM module, respectively, S denotes the SAM module, and the operation "$\times$" denotes point-wise multiplication.

The structure of RCSA does not significantly affect computing speed but improves the deblurring results to a certain extent. Experimental results are demonstrated in Sect. 4.

3.5 Loss Function

The coarse-to-fine approach desires that every mid-level output is the deblurred image of the corresponding scales. Thus, the training loss of the proposed UHDVD network is the MSE loss between the image content of the network output and the ground truth frame at each scale computed by

$$\mathcal{L}_{i\_MSE} = \sum_{s=1}^{S} \frac{\mathcal{K}_s}{C_i^s H_i^s W_i^s} ||D_i^s - G_i^s||_2^2, \tag{5}$$

where D_i^s and G_i^s are the deblurred image and the ground truth at the s-th scale of i-th frame, respectively; C_i^s,

(a) SRN [44] (SPS) 23.74/0.857 (b) SRN [44] (DPU) 24.08/0.866 (c) PSNR

(d) DMPHN [53] (SPS) 21.89/0.845 (e) DMPHN [53] (DPU) 22.58/0.861 (f) SSIM

Fig. 6 Quantitative evaluations of 4K image processing schemes (SPS and DPU) based on the 4KRD validation set

H_i^s, W_i^s are dimensions of multi-scale image; $\mathcal{K}_s$ is the weights for each scale. We empirically set $\mathcal{K}_{1,\,2,\,3,\,4} = [0.7,\ 0.15,\ 0.1,\ 0.05]$. In addition, S is the number of scales in our network, we set S to 4 in the paper. In addition, we add the Total Variation (TV) loss to avoid stripe artifact in the recovered image. The discrete definition of TV loss in i-th recovered frame is formulated as

$$\mathcal{L}_{i\_TV} = \sum_{p_{x,y} \in i} \sqrt{(p_{x+1,y} - p_{x,y})^2 + (p_{x,y+1} - p_{x,y})^2}, \tag{6}$$

where $p_{x,y} \in i$ denotes the pixel in i-th frame and x, y are the coordinates. So the total loss is formulated as

$$\mathcal{L}_{total} = \mathcal{L}_{i\_MSE} + \beta \mathcal{L}_{i\_TV}, \tag{7}$$

where β is set as $1e^{-7}$ to control the impact of TV loss. Note that our 4K real-time video deblurring network does not rely on other complicated loss functions such as adversarial loss (Nah et al., 2017; Kupyn et al., 2018) and optical flow loss (Pan et al., 2020), only using the MSE and TV loss can achieve competitive results as demonstrated in the next section.

4 Experiments

In this section, we evaluate the proposed algorithm on both synthetic datasets and real-world 4K videos with comparisons to the state-of-the-art image/video deblurring methods in terms of accuracy and visual effect. For fair comparisons, we also evaluate our method on public 720p and 2K datasets with these methods. The proposed 4K datasets[1] is available to the public for further discussion and research. More experimental results can be found in the supplementary material.

[1] 4KRD site: https://drive.google.com/drive/folders/19bjJLMgQkwIAQaZYvsUhEVaxzJQFwhHF?usp=sharing

4.1 Implementation Details

Our experiments are implemented in PyTorch and evaluated on a single NVIDIA Tesla V100 GPU with 32GB RAM. The batch size is set to 1 during training because every frame needs its previous deblurred frame as an extra feature. The Adam optimizer (Kingma & Ba, 2014) is used to train our models with patch size of 512×512. The initial learning rate is set to 0.0001 and the decay rate to 0.1. We normalize frames to the range of [0,1] and subtract 0.5 as in Nah et al. (2017), Zhang et al. (2019) since this pre-processing can speed up convergence and make better use of activation functions.

4.2 Dataset

As no public high-quality 4K dataset exists for image deblurring, we choose the scheme of Nah et al. (2019b) to generate a 4K Resolution Deblurring (4KRD) dataset. The proposed dataset covers a diversity of characters, people, artificial or natural objects, indoor scenes, outdoor landscapes, city street views, etc. There are two steps in generating this dataset: frame interpolation and dataset synthesis as in Nah et al. (2019). The video capturing equipments are mainstream flagship mobile phones, e.g., iPhone 11 Pro Max, HUAWEI Mate 30 Pro, and Samsung S20 Ultra. We also use a DJI Osmo Mobile 3 to stabilize mobile phones to make the captured videos as clear as possible.

High frame rates are necessary for the subsequent multi-frame fusion to ensure the continuity of frames in the synthetic dataset. However, we cannot directly capture 4K videos at high-frame-rates with smartphones due to hardware limitations. Therefore, we use the frame interpolation method Niklaus et al. (2017) to interpolate the recorded 4K videos from 30/60 fps to 480 fps as like the scheme of Nah et al. (2019b). Then we generate blurry frames by averaging a series of successive sharp frames. In addition to our 4K resolution dataset, we also use three public 720p deblurring datasets of GoPro (Nah et al., 2017), DVD (Su et al., 2017), and REDS (Nah et al., 2019) to test our UHDVD model. Specially, since the test ground truth is not available for the REDS (Nah et al., 2019) dataset, we select the validation set as our test data.

Although there is no public 4K resolution dataset at present, a 2K resolution dataset Slow-Flow (Janai et al., 2017) provides a Quad-HD real-world benchmark with ground-truth images. We evaluate all the 12 test video clips in the Slow-Flow dataset (Janai et al., 2017) and choose the blurry frames synthesized from 5 successive sharp frames.

4.3 Performance Evaluation

In this section, we evaluate our UHDVD method against the state-of-the-art video deblurring methods of Zhou et al.

Table 2 Quantitative evaluations against state-of-the-art deblurring methods on three 720p datasets, a 2K dataset (Janai et al., 2017), and our 4K 4KRD dataset

Methods	MSResNet (Nah et al., 2017)	SRN (Tao et al., 2018)	PSS-NSC (Gao et al., 2019)	DeblurGAN-v2 (Kupyn et al., 2019)	DMPHN (Zhang et al., 2019) Stack(4)/(1-2-4-8)	EDVR (Wang et al., 2019)	CDVD-TSP (Pan et al., 2020)	STFAN (Zhou et al., 2019)	Ours
720p GoPro (Nah et al., 2017)									
PSNR	28.45	30.10	31.58	29.55	31.20/30.25	26.87	**31.67**	28.63	31.38
SSIM	0.917	0.932	**0.948**	0.934	0.945/0.935	0.843	0.928	0.863	0.911
Time (ms)	747.8	731.7	1470	293.6	1029.3/30.9	384.6	4216.6	150.4	**25.3**
720p DVD (Su et al., 2017)									
PSNR	28.98	29.10	29.97	28.54	30.47/29.91	30.27	32.13	31.24	**32.21**
SSIM	0.885	0.899	0.919	0.925	0.881/0.866	0.917	0.927	0.934	**0.935**
Time (ms)	775.8	783.6	1360	312.2	987.9/30.4	289.2	4098.2	177.2	**27.6**
720p REDS (Nah et al., 2019)									
PSNR	26.49	25.40	26.98	25.61	25.18/25.06	**30.63⋆**	26.29	25.49	27.69
SSIM	0.742	0.734	0.814	0.731	0.724/0.724	**0.850⋆**	0.774	0.719	0.823
Time (ms)	802.6	823.3	1439	350.8	1069.9/29.3	325.7	3765.6	155.7	**26.9**
2K Slow-Flow (Janai et al., 2017)									
PSNR	27.70	28.81	–	28.75	28.29/27.94	26.42	28.64	28.23	**28.85**
SSIM	0.817	0.827	–	0.817	0.813/0.807	0.797	0.830	0.825	**0.831**
Time (ms)	2303.5	2677.1	–	833.6	3589.6/58.9	1302.7	7491.5	326.0	**38.5**
4K 4KRD									
PSNR	25.81	25.58	26.46	25.64	24.99/24.91	26.36	26.43	26.14	**28.17**
SSIM	0.778	0.759	0.801	0.763	0.757/0.748	0.803	0.793	0.800	**0.843**
Time (ms)	7543.4	8723.3	13546.7	3283.4	10378.1/399.4	2428.1	26922.9	953.2	**57.9**

The runtime (writing generated images to disk is not considered) is expressed in milliseconds of an image. We use bold and underline to indicate the best and the second-best performances, respectively. The ⋆ denotes that EDVR (Wang et al., 2019) uses the validation data for training on the REDS dataset

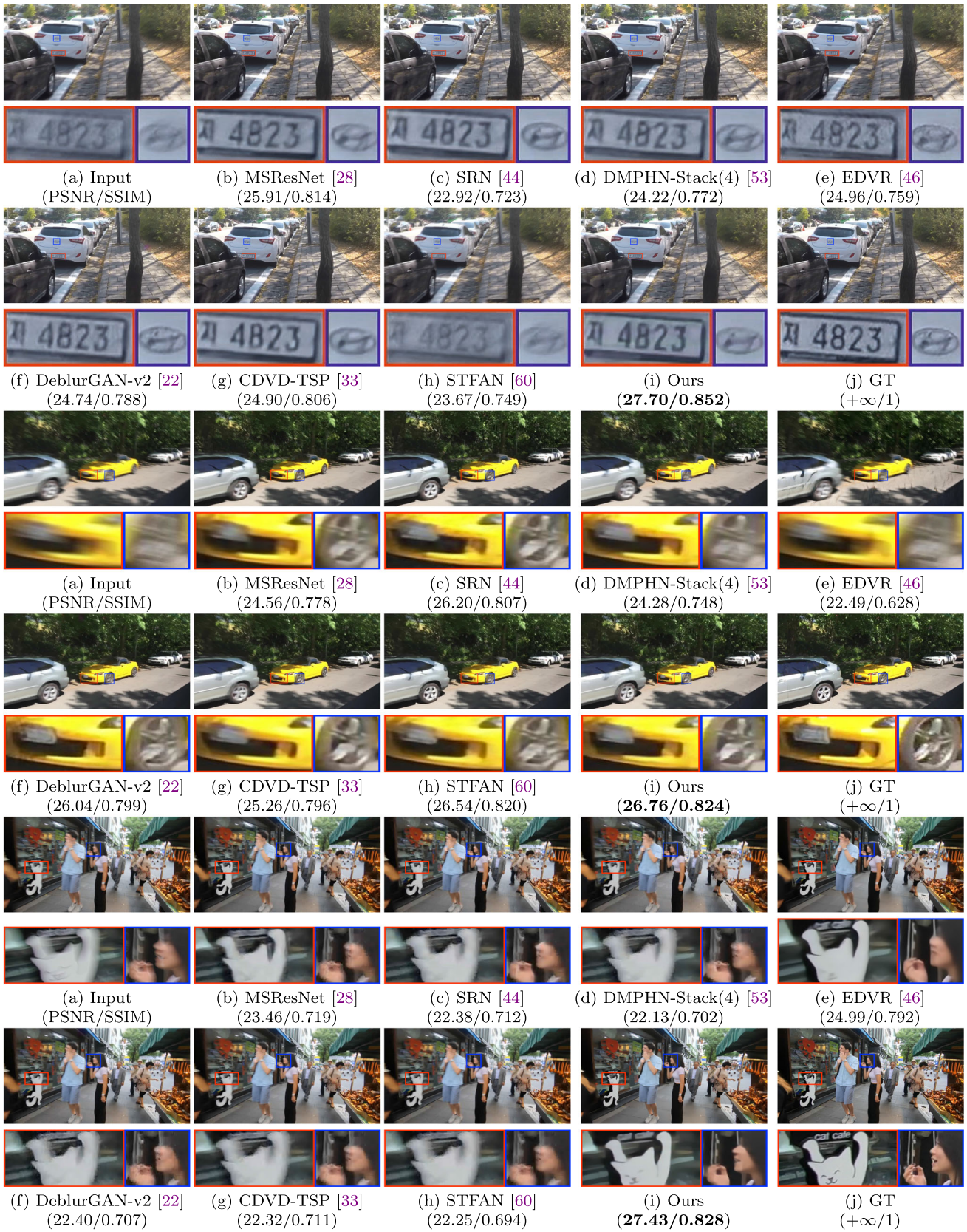

Fig. 7 Quantitative evaluations on different 720p datasets (top → bottom: GoPro (Nah et al., 2017), DVD (Su et al., 2017), REDS (Nah et al., 2019))

(2019), Su et al. (2017), Wang et al. (2019) and image deblurring approaches of Kupyn et al. (2019), Nah et al. (2017), Tao et al. (2018), Zhang et al. (2019). We evaluate these methods by three criteria: PSNR, SSIM, and average runtime of an image on each dataset. All these methods are tested in the same server environment.

As existing methods mainly processing the 720p images, we design two schemes for performance evluation. The first scheme is downsample-process-upsample (DPU) which processes an image at low-resolution and then up-sample the result. The other one is spitting-process-stitching (SPS) which splits an image into multi-pieces equally and stitches the processed pieces to the full-resolution. We compare the two schemes on our 4KRD validation dataset. As shown in Fig. 6, SPS has serious artifacts because it only uses the local region on each piece. Meanwhile, scheme DPU has higher PSNR/SSIM than SPS. Therefore, SPS strategy is deserted in our subsequent experiments.

Quantitative Evaluation. Table 2 shows that the proposed method performs favorably against the state-of-the-art algorithms on the five datasets: GoPro (Nah et al., 2017), DVD (Su et al., 2017), REDS (Nah et al., 2019), Slow-Flow (Janai et al., 2017) and 4KRD. The run time of all the methods reported in this table is based on the same test environment and hardware (Tesla V100 GPU with 32GB RAM).

On the DVD benchmark (Su et al., 2017), Slow-Flow (Janai et al., 2017) dataset, and our 4KRD dataset, our algorithm obtain the best results in terms of PSNR and SSIM, while on the REDS (Nah et al., 2019) dataset, we are also the suboptimal method. Although EDVR (Wang et al., 2019) achieves the best results on the REDS dataset, we note that this method uses all the validation videos of REDS to train their model. In addition, since CDVD-TSP (Pan et al., 2020) explicitly uses temporal information of multiple frames, this method exceeds us on the GoPro dataset. PSS-NSC (Gao et al., 2019) also achieves the highest SSIM value on the GoRro dataset based on a generic principled parameter selective sharing scheme. However, our algorithm is 166× and 58× faster than CDVD-TSP (Pan et al., 2020) and PSS-NSC (Gao et al., 2019), respectively. Fig. 7 shows three visual examples from the GoPro (Nah et al., 2017), DVD (Su et al., 2017) and REDS (Nah et al., 2019) datasets, respectively. Figure 8 shows one example from the 2K Slow-Flow dataset. We are the best in PSNR and the method DeblurGAN-v2 Kupyn et al. (2019) is the second. However, there are some purple shadow regions in its left-bottom region of Fig. 8d, and this artifact is present throughout the video clip generated by Kupyn et al. (2019) (Fig. 6). Figure 9 gives two examples from our 4KRD dataset. Our method achieved better results in all of these visual effects.

Qualitative Evaluation. To further validate the generalization ability of our network, we also qualitatively compare the proposed network with other algorithms on real blurry video clips on our 4K real test videos. As illustrated in Fig. 10, the proposed method can restore clearer results with more details than other methods. These comparison results show that our UHDVD method can robustly handle real blurs in most scenes. For example, the head region of the first image, the characters of the first two images, and the kiosk lattice in the last image contain sharper structures and details than the results generated by other approaches.

Additionally, to validate the performance of the proposed UHDVD in large motion blurs specifically, we further compare our model with above methods and a new method ESTRNN (Zhong et al., 2020) which has a similar architecture in the temporal feature extraction branch. The visual results of the cyclist video on the DVD test dataset (Su et al., 2017) are shown in Fig. 13. As observed, our algorithm generates sharper details in the handlebar and right leg regions. Compared with others, our deblurred results have fewer artifacts around objects.

4.4 Effectiveness of TFA

To fully exploit the continuity between video frames, we proposed the Temporal Feature Attention module described in Sect. 3.1 to extract the temporal features of three previous deblurred frames and the current frame simultaneously. To demonstrate the effectiveness of the TFA module and its performances with different configurations, we conduct a series of contrast experiments as follows: i) A model without a TFA module which just concatenates current blurry frame and one previous deblurred frame at channel as the initial input; ii) Some models with different numbers of previous/future blurry frames or previous deblurred frames as the initial input; iii) Some models with different numbers of RDBs ("RDB_c32" in Fig. 3) in the TFA module. All the above models are trained with the same encoder–decoder branch at the same epoch.

The quantitative evaluation results of i) are shown in Table 3 and the visual results are shown in Fig. 12. As shown, we can see that our model with the TFA module is able to yield better results on PSNR and image details of moving objects than those without the TFA module. The quantitative results (trained at the same epoch) of ii) and iii) are shown in Table 4. Obviously, more input frames (no matter blurry or deblurred) and RDBs will decrease the efficiency of our UHDVD model. So the number of input frames and RDBs is a trade-off between efficiency and accuracy. While situation "P0F0D3" achieves a similar PSNR to "P2F2D1" at a faster speed, then the benchmark model is based on "P0F0D3". Meanwhile, we choose 3 as the default value for the number of RDBs.

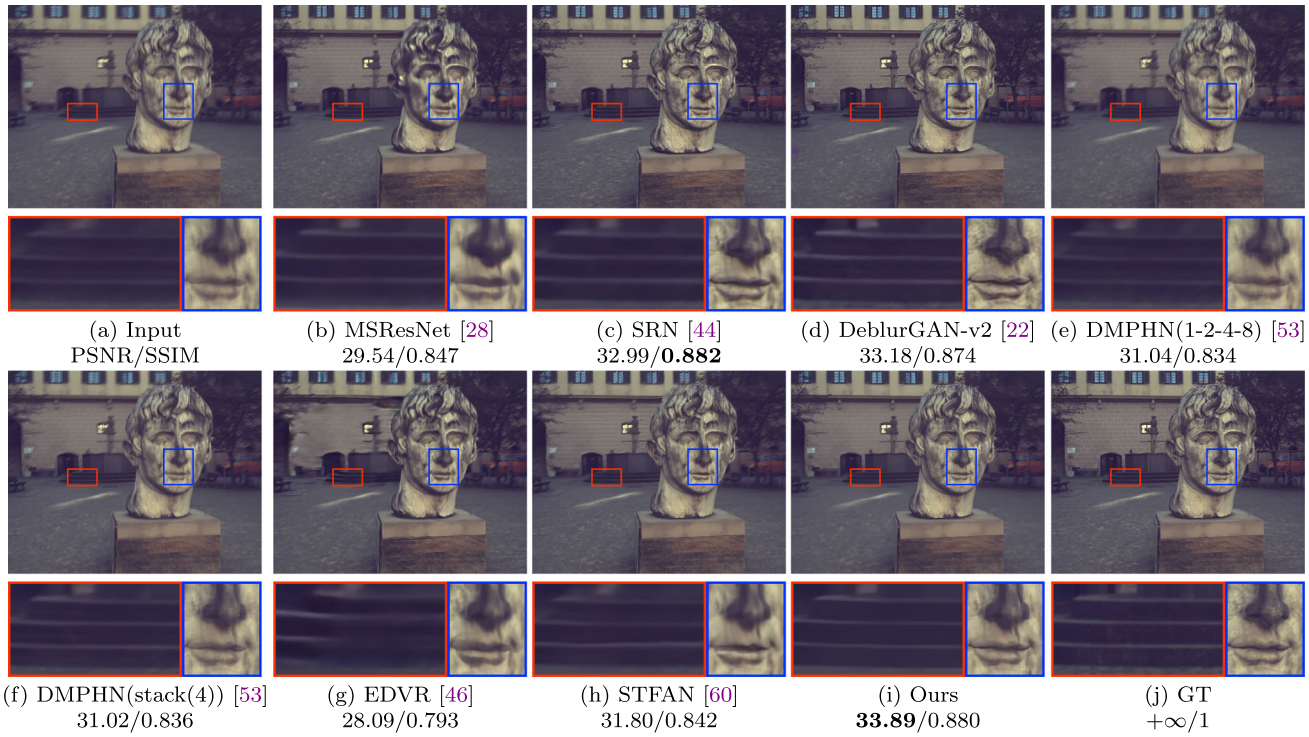

Fig. 8 Quantitative evaluation on Slow-Flow dataset with other state-of-the-art methods (zoom in for best view)

4.5 Effectiveness of SPAA

To validate the effectiveness of Separable-Patch Acceleration Architecture (SPAA), we conduct experiments on random 1000 blurry frames of 720p, 2K, and 4K resolutions, respectively. The results in Table 5 show that we have more than doubled the computing speed by using the proposed acceleration architecture, while the PSNRs are almost the same. These results demonstrate that the proposed separable-patch acceleration architecture is pivotal to increasing speed and enabling 4K image deblurring in real-time. Meanwhile, we compare the FLOPS indicator of our UHDVD model with and without the SPAA module in the same above environment. The results are shown in Table 5. As shown, the FLOPS has fallen 1.1T by using the SPAA module on 4K images. Although the proposed SPAA is seemingly simple, the parallel process of several patches at the same time is the goal of SPAA and it has been shown that this can reduce operation time significantly.

4.6 Effectiveness of RCSA

To validate the effectiveness of the Residual Channel-Spatial Attention (RCSA) module of our network, we also trained a new model without the whole RCSA module on the 4KRD dataset. The baseline model only uses two layers of the residual block without any CAM and SAM. Except for this difference, everything else is exactly the same as the initial model. The quantitative results are shown in Table 6. It indicates that our UHDVD model achieves 0.5 dB gain than the model without using RCSA in terms of PSNR. Meanwhile, the calculation speed of the two models is almost the same by using the separable-patch acceleration pipeline. Some qualitative results are shown in Fig. 11. The qualitative results also demonstrate the effectiveness of the RCSA module.

4.7 Model Size, FLOPS and Run Time

Our model achieves better efficiency and accuracy results, while we also have the smallest model size. Table 7 shows the model size of all evaluated methods. The proposed UHDVD model has the smallest model size among them, which is also an important factor in our speed up on processing. Additionally, we also compare the FLOPS ($\times 10^{12}$) of our UHDVD model with other methods on 4K data, the results are also shown in Table 7. The FLOPS of UHDVD is also the smallest.

The proposed UHDVD can process a 2160×3840 image within 30 ms without the TFA module, which means our model supports real-time 4K video deblurring task at 35fps. DMPHN (Zhang et al., 2019) has also reached real-time deblurring on images of 720p resolution in their high-

Fig. 9 Quantitative evaluations on our 4K resolution deblurring datasets. Our UHDVD generates much clearer images with higher PSNR and SSIM (zoom in for best view)

efficiency version. From the quantitative results of DMPHN in Table 2, it can be observed that their high-efficiency version (without stack) yields lower PSNR than the one with stack in all test datasets. Additionally, their high-efficiency version still does not work to reach real-time on 4K resolution. As shown in Table 3, the high-efficient UHDVD (without TFA module) is 10× faster than the method DMPHN-(1-2-4-8) on 4K resolution videos. Furthermore, our model also improves the operational efficiency on 720p and reaches the speed of 12.7 ms per frame. It should be pointed out that we follow the prototype in Zeng et al. (2020), Zhang et al. (2019), the time we considered is the GPU process time, there are runtime overheads related to I/O operations, which are directly proportional to the size of images. So the real-time processing means GPU real-time in the strict sense (Figs. 12, 13).

The following factors contribute to our speed up: i) multi-scale scheme reduces the input image size of the first three scales; ii) multi-patch and separable-patch acceleration architecture increase the speed of calculation; iii) relatively few network layers and parameter amounts.

5 Conclusion

In this paper, we propose a 4K video real-time deblurring network using an asymmetrical encoder–decoder architecture. We integrate multi-scale and multi-patch schemes in a unified framework to improve efficiency and accuracy simultaneously, and adopt the TFA module based on residual dense blocks to fully use the temporal feature between video frames. In contrast to prior work, we use the asymmetrical

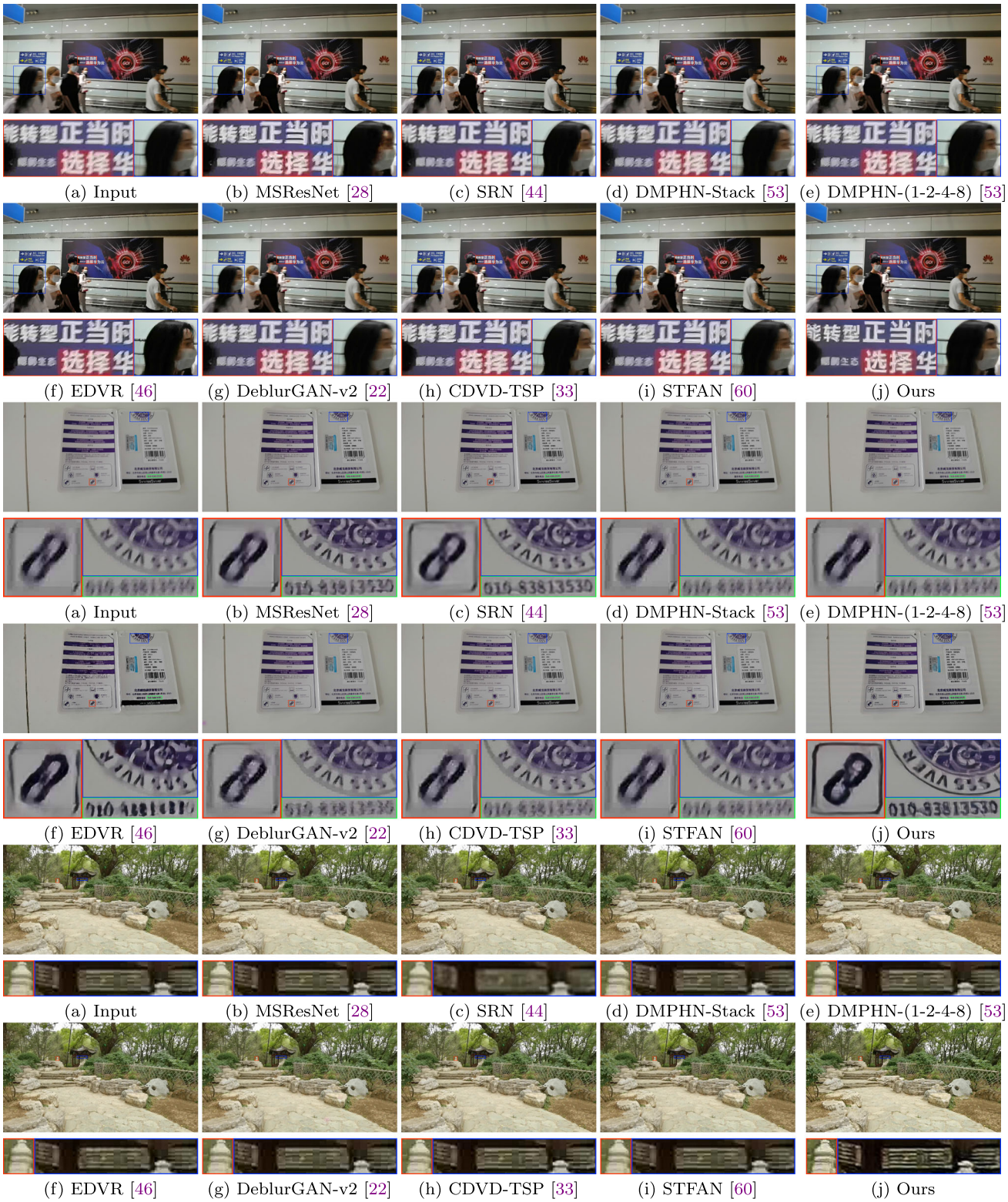

Fig. 10 Qualitative evaluations on our 4KRD real test dataset. The proposed UHDVD model generates much clearer results in both detail and full image (zoom in for best view)

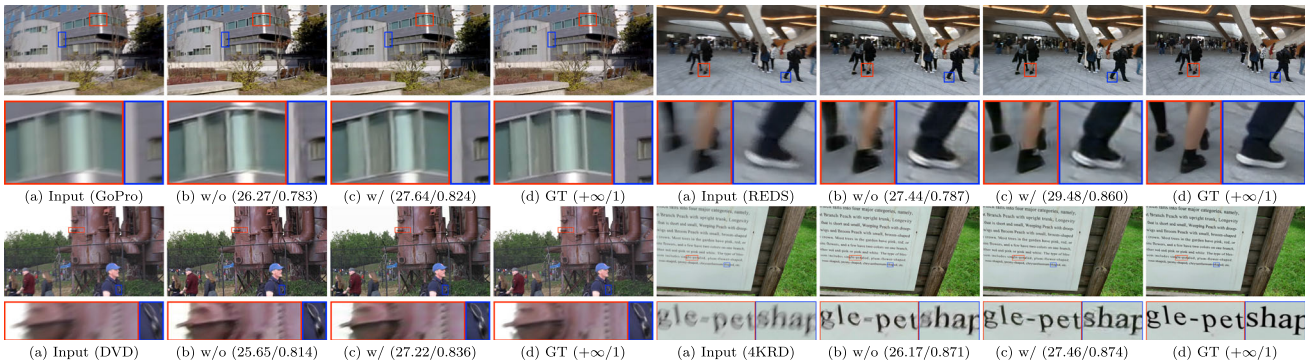

Fig. 11 Quantitative evaluations on different datasets with (w/) the whole RCSA module or not (w/o)

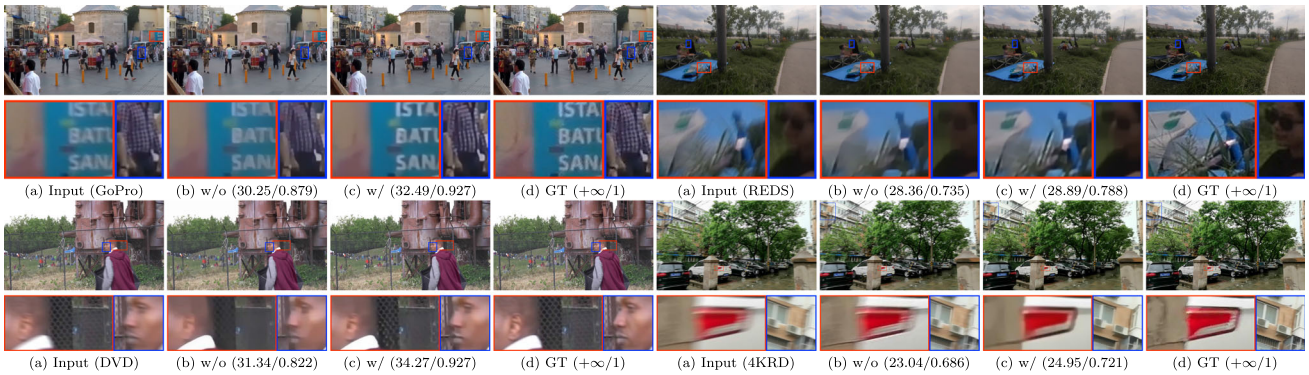

Fig. 12 Quantitative evaluations on different datasets with (w/) the whole TFA module or not (w/o)

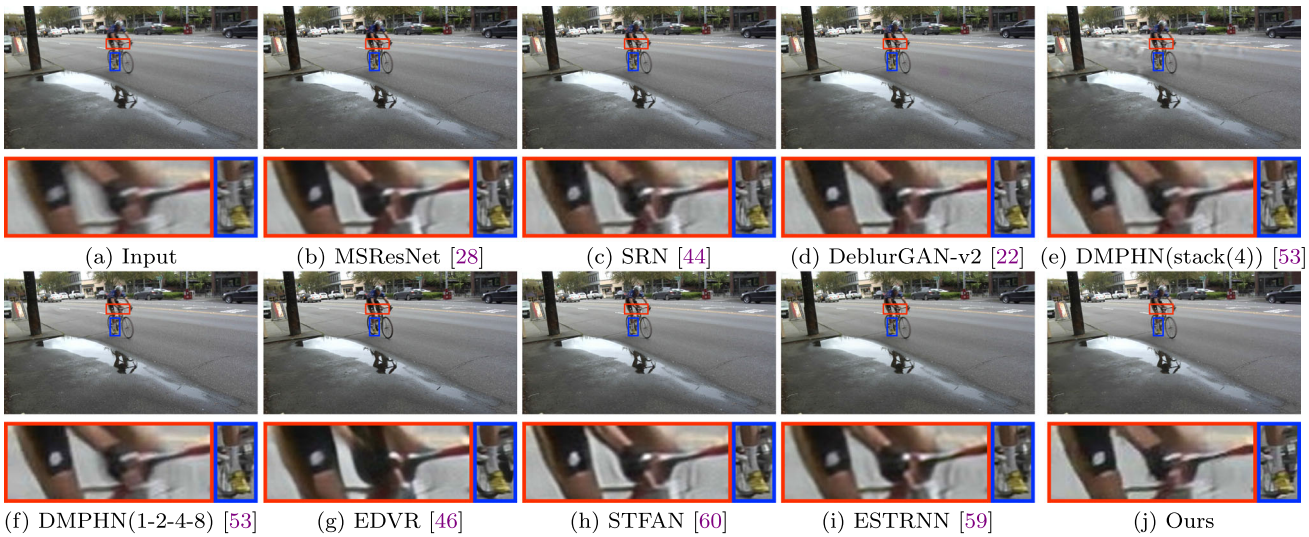

Fig. 13 Deblurred results of cyclist (bicycle) video on DVD real test (zoom in for best view)

Table 3 PSNR/Time(ms) of our UHDVD model on different datasets with (w/) the TFA module or not (w/o)

Datasets	GoPro	DVD	REDS	Slow-Flow	4KRD
w/o TFA	31.33/12.7	32.19/13.2	27.53/13.9	28.61/21.3	27.88/27.9
w/ TFA	31.38/25.3	32.21/27.6	27.69/26.9	28.85/38.5	28.17/57.9

Table 4 TFA module with different configurations

	P1F1D1	P2F2D1	P3F3D1	P0F0D2	P0F0D3	RDBs1	RDBs2	RDBs3	RDBs4
PSNR	27.51	27.68	27.72	27.52	27.66	27.28	27.54	27.66	27.71
Time (ms)	26.1	32.2	43.4	22.7	25.3	18.2	21.7	25.3	35.4

Left: "P#F#D#" denotes the number of Previous/Future blurry frames or previous Deblurred frames; Right: "RDBs#" denotes the number of RDBs

Table 5 PSNR/Time(ms) and TFLOPS of our UHDVD with (w/) the SPAA module or not (w/o) on different resolution

	720p	2K	4K	TFLOPS
w/o SPAA	28.83/41.1	28.33/59.5	27.88/80.6	2.9
w/ SPAA	28.79/27.1	28.34/39.1	27.87/58.3	1.8

Table 6 PSNR/Time(ms) of our UHDVD on different datasets with (w/) the RCSA module or not (w/o)

Datasets	GoPro	DVD	REDS	Slow-Flow	4KRD
w/o RCSA	30.64/12.4	31.57/12.3	26.64/12.5	28.67/18.4	27.38/22.6
w/ RCSA	31.33/12.7	32.19/13.2	27.53/13.9	29.03/22.3	27.88/27.9

Table 7 Model size and TFLOPS of our UHDVD model and other methods on 4K images

Methods	DMPHN (Zhang et al., 2019) Stack(4)/(1-2-4-8)	MSResNet (Nah et al., 2017)	SRN (Tao et al., 2018)	EDVR (Wang et al., 2019)	DeblurGAN-v2 (Kupyn et al., 2019)	CDVD-TSP (Pan et al., 2020)	ESTRNN (Zhong et al., 2020)	STFAN (Zhou et al., 2019)	Ours
Model Size (Mb)	86.9/29.0	303.6	33.6	23.6	15.0	16.2	5.2	5.4	**5.0**
TFLOPS	14.8/5.1	15.8	12.9	10.8	6.1	/	1.9	2.1	**1.8**

encoder–decoder structure to build our network with fewer convolution layers to save calculation costs. In addition, we adopt the separable-patch acceleration architecture to reach the real-time processing speed at 35 fps on 4K resolution videos without the TFA module. For ultra high-definition deblurring, we construct a dataset containing images of 4K resolution. Quantitative and qualitative results show that the proposed method performs favorably against the state-of-the-art deblurring methods on synthetic and real-world datasets with 720p, 2K, and 4K resolutions.

Acknowledgements This research was funded in part by National Natural Science Foundation of China (NSFC) under Grant 62322216 and 62172409, and funded in part by Shenzhen Science and Technology Program (Grant No. RCYX20221008092849068, No. JCYJ20220818102012025, No. JCYJ20220530145209022).

References

Bar, L., Berkels, B., Rumpf, M., & Sapiro, G. (2007). A variational framework for simultaneous motion estimation and restoration of motion-blurred video. In *IEEE international conference on computer vision*.

Chen, L., Fang, F., Wang, T., & Zhang, G. (2019). Blind image deblurring with local maximum gradient prior. In *IEEE conference on computer vision and pattern recognition*.

Chen, L. C., Papandreou, G., Schroff, F., & Adam, H. (2017). *Rethinking atrous convolution for semantic image segmentation*. arXiv preprint arXiv:1706.05587

Cho, S., Matsushita, Y., & Lee, S. (2007). Removing non-uniform motion blur from images. In *IEEE international conference on computer vision*.

Deng, S., Ren, W., Yan, Y., Wang, T., Song, F., & Cao, X. (2021). Multi-scale separable network for ultra-high-definition video deblurring. In *IEEE international conference on computer vision* (pp. 14030–14039).

Dong, W., Zhang, L., Shi, G., & Wu, X. (2011). Image deblurring and super-resolution by adaptive sparse domain selection and adaptive regularization. *IEEE Transactions on Image Processing, 20*(7), 1838–1857.

Gao, H., Tao, X., Shen, X., & Jia, J. (2019). Dynamic scene deblurring with parameter selective sharing and nested skip connections. In *IEEE conference on computer vision and pattern recognition*.

Gong, D., Yang, J., Liu, L., Zhang, Y., Reid, I., Shen, C., Van Den Hengel, A., & Shi, Q. (2017). From motion blur to motion flow: A deep learning solution for removing heterogeneous motion blur. In *IEEE conference on computer vision and pattern recognition* (pp. 2319–2328).

Hu, X., Ren, W., Yu, K., Zhang, K., Cao, X., Liu, W., & Menze, B. (2021). Pyramid architecture search for real-time image deblurring. In *IEEE international conference on computer vision*.

Hu, Z., & Yang, M. H. (2015). Learning good regions to deblur images. *International Journal of Computer Vision, 115*(3), 66.

Hyun Kim, T., Ahn, B., & Mu Lee, K. (2013). Dynamic scene deblurring. In *IEEE international conference on computer vision* (pp. 3160–3167).

Hyun Kim, T., & Mu Lee, K. (2014). Segmentation-free dynamic scene deblurring. In*IEEE conference on computer vision and pattern recognition* (pp. 2766–2773).

Hyun Kim, T., Mu Lee, K., Scholkopf, B., & Hirsch, M. (2017). Online video deblurring via dynamic temporal blending network. In *IEEE international conference on computer vision* (pp. 4038–4047).

Janai, J., Guney, F., Wulff, J., Black, M. J., & Geiger, A. (2017). Slow flow: Exploiting high-speed cameras for accurate and diverse optical flow reference data. In *IEEE conference on computer vision and pattern recognition*.

Ji, H., & Wang, K. (2012). A two-stage approach to blind spatially-varying motion deblurring. In *IEEE conference on computer vision and pattern recognition*.

Jiang, Z., Zhang, Y., Zou, D., Ren, J., Lv, J., & Liu, Y. (2020). Learning event-based motion deblurring. In *IEEE conference on computer vision and pattern recognition*.

Keskar, N. S., Mudigere, D., Nocedal, J., Smelyanskiy, M., & Tang, P. T. P. (2016). On large-batch training for deep learning: Generalization gap and sharp minima. In *International conference on learning representations*.

Kim, S. Y., Oh, J., & Kim, M. (2019). Deep sr-itm: Joint learning of super-resolution and inverse tone-mapping for 4k uhd hdr applications. In *IEEE international conference on computer vision*.

Kingma, D. P., & Ba, J. (2014). *Adam: A method for stochastic optimization*. arXiv preprint arXiv:1412.6980.

Krishnan, D., Tay, T., & Fergus, R. (2011). Blind deconvolution using a normalized sparsity measure. In *IEEE conference on computer vision and pattern recognition*.

Kupyn, O., Budzan, V., Mykhailych, M., Mishkin, D., & Matas, J. (2018). Deblurgan: Blind motion deblurring using conditional adversarial networks. In *IEEE conference on computer vision and pattern recognition* (pp. 8183–8192).

Kupyn, O., Martyniuk, T., Wu, J., & Wang, Z. (2019). Deblurgan-v2: Deblurring (orders-of-magnitude) faster and better. In *IEEE international conference on computer vision*.

Lai, W. S., Ding, J. J., Lin, Y. Y., & Chuang, Y. Y. (2015). Blur kernel estimation using normalized color-line prior. In *IEEE conference on computer vision and pattern recognition*.

Li, L., Pan, J., Lai, W.S., Gao, C., Sang, N., & Yang, M. H. (2018). Learning a discriminative prior for blind image deblurring. In *IEEE conference on computer vision and pattern recognition*.

Liu, Z., Yeh, R. A., Tang, X., Liu, Y., & Agarwala, A. (2017). Video frame synthesis using deep voxel flow. In *IEEE international conference on computer vision* (pp. 4463–4471).

Michaeli, T., & Irani, M. (2014). Blind deblurring using internal patch recurrence. In *European conference on computer vision*.

Nah, S., Baik, S., Hong, S., Moon, G., Son, S., Timofte, R., & Mu Lee, K. (2019). Ntire 2019 challenge on video deblurring and super-resolution: Dataset and study. In *IEEE conference on computer vision and pattern recognition workshops*.

Nah, S., Hyun Kim, T., & Mu Lee, K. (2017). Deep multi-scale convolutional neural network for dynamic scene deblurring. In *IEEE conference on computer vision and pattern recognition* (pp. 3883–3891).

Nah, S., Son, S., & Lee, K. M. (2019). Recurrent neural networks with intra-frame iterations for video deblurring. In *IEEE conference on computer vision and pattern recognition* (pp. 8102—8111).

Nah, S., Timofte, R., Baik, S., Hong, S., Moon, G., Son, S., & Mu Lee, K. (2019). Ntire 2019 challenge on video deblurring: Methods and results. In *IEEE conference on computer vision and pattern recognition workshops*.

Nan, Y., Quan, Y., & Ji, H. (2020). Variational-em-based deep learning for noise-blind image deblurring. In *IEEE conference on computer vision and pattern recognition*.

Niklaus, S., Mai, L., & Liu, F. (2017). Video frame interpolation via adaptive convolution. In *IEEE conference on computer vision and pattern recognition* (pp. 670–679).

Pan, J., Bai, H., & Tang, J. (2020). Cascaded deep video deblurring using temporal sharpness prior. In *IEEE conference on computer vision and pattern recognition* (pp. 3043–3051).

Parmar, N., Vaswani, A., Uszkoreit, J., Kaiser, Ł., Shazeer, N., Ku, A., & Tran, D. (2018). *Image transformer*. arXiv preprint arXiv:1802.05751

Perrone, D., & Favaro, P. (2014). Total variation blind deconvolution: The devil is in the details. In *IEEE conference on computer vision and pattern recognition*.

Ren, W., Pan, J., Cao, X., & Yang, M. H. (2017). Video deblurring via semantic segmentation and pixel-wise non-linear kernel. In *IEEE international conference on computer vision* (pp. 1077–1085).

Schuler, C. J., Hirsch, M., Harmeling, S., & Schölkopf, B. (2015). Learning to deblur. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 38*(7), 1439–1451.

Shan, Q., Jia, J., & Agarwala, A. (2008). High-quality motion deblurring from a single image. *ACM Transactions on Graphics, 27*(3), 1–10.

Su, S., Delbracio, M., Wang, J., Sapiro, G., Heidrich, W., & Wang, O. (2017). Deep video deblurring for hand-held cameras. In *IEEE conference on computer vision and pattern recognition* (pp. 1279–1288).

Suin, M., Purohit, K., & Rajagopalan, A. (2020). Spatially-attentive patch-hierarchical network for adaptive motion deblurring. In *IEEE conference on computer vision and pattern recognition* (pp. 3606–3615).

Sun, J., Cao, W., Xu, Z., & Ponce, J. (2015). Learning a convolutional neural network for non-uniform motion blur removal. In *IEEE conference on computer vision and pattern recognition* (pp. 769–777).

Sun, L., Cho, S., Wang, J., & Hays, J. (2013). Edge-based blur kernel estimation using patch priors. In *IEEE international conference on computational photography*.

Tao, X., Gao, H., Liao, R., Wang, J., & Jia, J. (2017). Detail-revealing deep video super-resolution. In *IEEE international conference on computer vision* (pp. 4472–4480).

Tao, X., Gao, H., Shen, X., Wang, J., & Jia, J. (2018). Scale-recurrent network for deep image deblurring. In *IEEE conference on computer vision and pattern recognition* (pp. 8174–8182).

Wang, P., Chen, P., Yuan, Y., Liu, D., Huang, Z., Hou, X., & Cottrell, G. (2018). Understanding convolution for semantic segmentation. In IEEE winter conference on applications of computer vision (pp. 1451–1460).

Wang, X., Chan, K. C., Yu, K., Dong, C., & Change Loy, C. (2019). Edvr: Video restoration with enhanced deformable convolutional networks. In *IEEE conference on computer vision and pattern recognition workshops*.

Wieschollek, P., Hirsch, M., Scholkopf, B., & Lensch, H. P. A. (2017). Learning blind motion deblurring. In *IEEE international conference on computer vision*.

Wulff, J., & Black, M. J. (2014). Modeling blurred video with layers. In *European conference on computer vision* (pp. 236–252).

Xu, L., Zheng, S., & Jia, J. (2013). Unnatural l0 sparse representation for natural image deblurring. In *IEEE conference on computer vision and pattern recognition*.

Yu, F., & Koltun, V. (2015). *Multi-scale context aggregation by dilated convolutions*. arXiv preprint arXiv:1511.07122

Zamir, S. W., Arora, A., Khan, S., Hayat, M., Khan, F. S., Yang, M. H., & Shao, L. (2021). *Multi-stage progressive image restoration*. arXiv preprint arXiv:2102.02808

Zeng, H., Cai, J., Li, L., Cao, Z., & Zhang, L. (2020). Learning image-adaptive 3d lookup tables for high performance photo enhancement in real-time. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 6*, 66.

Zhang, H., Dai, Y., Li, H., & Koniusz, P. (2019). Deep stacked hierarchical multi-patch network for image deblurring. In *IEEE conference on computer vision and pattern recognition* (pp. 5978–5986).

Zhang, J., Pan, J., Ren, J., Song, Y., Bao, L., Lau, R. W., & Yang, M. H. (2018). Dynamic scene deblurring using spatially variant recurrent neural networks. In *IEEE conference on computer vision and pattern recognition*.

Zhang, K., Luo, W., Zhong, Y., Ma, L., Liu, W., & Li, H. (2018). Adversarial spatio-temporal learning for video deblurring. *IEEE Transactions on Image Processing, 28*(1), 291–301.

Zhang, K., Luo, W., Zhong, Y., Ma, L., Stenger, B., Liu, W., & Li, H. (2020). Deblurring by realistic blurring. In *IEEE conference on computer vision and pattern recognition* (pp. 2737–2746).

Zhang, K., Zuo, W., & Zhang, L. (2019). Deep plug-and-play super-resolution for arbitrary blur kernels. In *IEEE conference on computer vision and pattern recognition*.

Zhao, Z., Xiong, B., Gai, S., & Wang, L. (2020). Improved deep multi-patch hierarchical network with nested module for dynamic scene deblurring. *IEEE Access, 8*, 62116–62126.

Zhong, Z., Gao, Y., Zheng, Y., & Zheng, B. (2020). Efficient spatio-temporal recurrent neural network for video deblurring. In *European conference on computer vision* (pp. 191–207). Springer.

Zhou, S., Zhang, J., Pan, J., Xie, H., Zuo, W., & Ren, J. (2019). Spatio-temporal filter adaptive network for video deblurring. In *IEEE international conference on computer vision* (pp. 2482–2491).

Zhou, Y., & Komodakis, N. (2014). A map-estimation framework for blind deblurring using high-level edge priors. In *European conference on computer vision*.