

# Real-world Video Deblurring: A Benchmark Dataset and An Efficient Recurrent Neural Network

Zhihang Zhong · Ye Gao · Yinqiang Zheng · Bo Zheng · Imari Sato

Received: date / Accepted: date

**Abstract** Real-world video deblurring in real time still remains a challenging task due to the complexity of spatially and temporally varying blur itself and the requirement of low computational cost. To improve the network efficiency, we adopt residual dense blocks into RNN cells, so as to efficiently extract the spatial features of the current frame. Furthermore, a global spatio-temporal attention module is proposed to fuse the effective hierarchical features from past and future frames to help better deblur the current frame. Another issue that needs to be addressed urgently is the lack of a real-world benchmark dataset. Thus, we contribute a novel dataset (BSD) to the community, by collecting paired blurry/sharp video clips using a co-axis beam splitter acquisition system. Experimental results show that the proposed method (ESTRNN) can achieve better deblurring performance both quantitatively and qualitatively with less computational cost against state-of-the-art video deblurring methods. In addition, cross-validation experiments between datasets illustrate the high generality of BSD over the synthetic datasets. The code and dataset are released at <https://github.com/zzh-tech/ESTRNN>.

---

Zhihang Zhong  
 The University of Tokyo, Japan

Ye Gao  
 Tokyo Research Center, Huawei, Japan

Yinqiang Zheng  
 The University of Tokyo, Japan  
 E-mail: yqzheng@ai.u-tokyo.ac.jp

Bo Zheng  
 Tokyo Research Center, Huawei, Japan

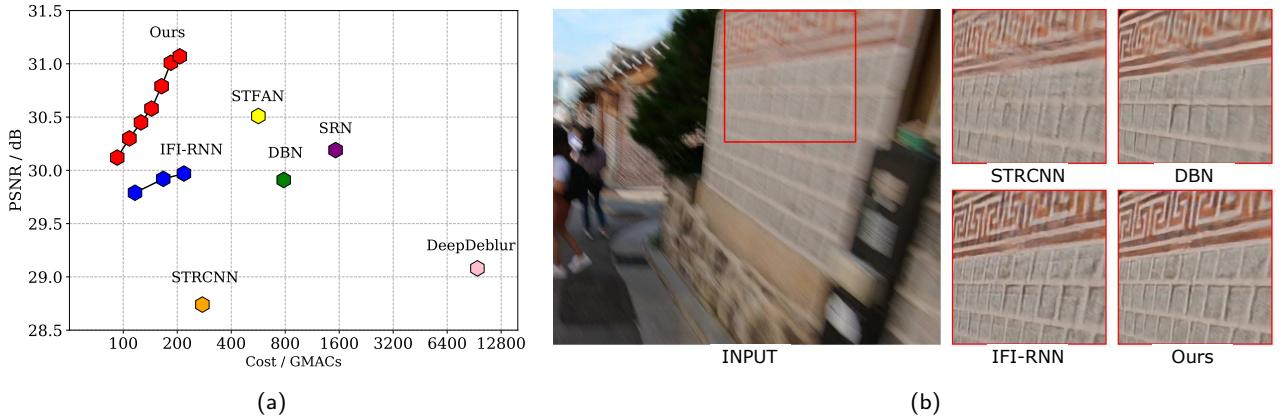
Imari Sato  
 National Institute of Informatics, Japan

**Keywords** Video deblurring · Network efficiency · RNN · Real-world dataset · Beam-splitter acquisition system

## 1 Introduction

Nowadays, video recording usually suffers from the quality issues caused by motion blur. This is especially true in poorly illuminated environment, where one has to lengthen the exposure time for sufficient brightness. A great variety of video deblurring methods have been proposed, which have to deal with two competing goals, i.e., to improve the deblurring quality and to reduce the computational cost. Reducing computational cost while ensuring deblurring quality is of critical importance for low-power mobile devices, such as smartphones.

To properly make use of the spatio-temporal correlation of the video signal is the key to achieve better performance on video deblurring. Deep learning-based methods have brought great advances in the field of video deblurring. The CNN-based methods [37, 42] make an inference of the deblurred frame by stacking neighboring frames with current frame as input to the CNN framework. The RNN-based methods, like [16, 27, 44, 56], employ recurrent neural network architecture to transfer the effective information frame by frame for deblurring. However, how to utilize spatio-temporal dependency of video for deblurring more efficiently still needs to be explored. The CNN-based methods are usually cumbersome in dealing with spatio-temporal dependency of concatenated neighboring frames, and the existing RNN-based methods have limited capacity to transfer the effective information temporally. Thus, they either suffer from huge computational cost, or ineffectiveness of deblurring.



**Fig. 1** A comparison of network efficiency on video deblurring. SRN [41], DeepDeblur [26] are methods for image deblurring, and STRCNN [16], DBN [37], IFI-RNN [27] are methods for video deblurring. (a) shows the computational cost required for processing a frame of 720P(1280 × 720) video and the corresponding performance of each model on GOPRO [26] dataset in terms of GMACs and PSNR, respectively. (b) shows the deblurred image generated by SoTA video deblurring methods and ours.

The quality of the benchmark dataset is also critical to the deblurring performance of the data-driven methods. It is worth noting that all mainstream deblurring datasets [25, 26, 37] are synthesized by averaging high-fps videos. Unnatural data distributions and artifacts in synthetic blur will inevitably lead to poor generalization of the models that are trained on synthetic datasets, especially for real blurry images/videos. The community is currently in urgent need of a real-world deblurring dataset. However, obtaining the corresponding ground truth for the captured blurry image/video remains an unsolved challenge.

To optimize the trade-off between computational cost and deblurring performance, we propose an efficient spatio-temporal recurrent neural network, denoted by ESTRNN. In this work, we mainly focus on the network efficiency, which directly reflects on the deblurring performance of the method with limited computational resources. To make a more computationally efficient video deblurring method, we develop our method through amelioration of basic RNN architecture from three aspects: (i) In temporal domain, the high-level features generated by RNN cell are more informative, which are more suitable for temporal feature fusion (see Fig. 2) than using channel-concatenated neighboring frames as input. Reusing high-level features of neighboring frames can also help to improve the overall network efficiency; (ii) It is obvious that not all high-level features from neighboring frames are beneficial to deblurring of the current frame. Thus, it is worth designing an attention module [1] that allows the method to focus on more informative parts of the high-level features from other frames. To this end, we propose a novel global spatio-temporal attention

module (see Sec. 3.3) for efficient temporal feature fusion; (iii) Regarding spatial domain, how to extract the spatial features from the current frame will affect the quality of information transmitted in temporal domain. In other words, well generated spatial features of each frame are a prerequisite for ensuring good temporal feature fusion. Therefore, we integrate the residual dense block [51] as backbone into RNN cell to construct RDB cell (see Sec. 3.2). The high-level hierarchical features generated by RDB cell are more computationally efficient with richer spatial information. With the above improvement, the proposed method can achieve better performance with less computational cost against SoTA deblurring methods, as illustrated in Fig. 1(a). Due to making full use of spatio-temporal dependency of video signal, our method is exceptionally good at restoring high-frequency details of the blurry frame compared with SoTA video deblurring methods, as shown in Fig. 1(b).

Furthermore, to get out of the predicament of lacking real-world deblurring dataset, we design a co-axis beam splitter acquisition system for data sample collection. Two cameras with distinct exposure schemes are co-axially aligned via a beam splitter to capture both blurry and sharp video of the same scene simultaneously. Empowered by the proposed beam-splitter acquisition system, we contribute the first real-world video deblurring dataset (BSD) to this field. BSD includes three different blur intensity configurations in dynamic scenes and contains various ego-motion and object-motion types. The advantages of the collected real-world dataset are verified by cross-validation experiments with synthetic datasets. Specifically, models trained on BSD can obtain migratory deblurring ca-

pabilities for other datasets; in contrast, the models trained on synthetic datasets have very limited generality, even on other synthetic datasets.

Our major contributions can be summarized as follows:

- To better exploit the spatio-temporal correlation of video signal for deblurring, we propose a novel RNN-based model that uses a recurrent cell based on residual dense blocks and a global spatio-temporal attention module to generate hierarchical spatial features and fuse the high-level features of neighboring frames, respectively.
- We design a beam splitter acquisition system to capture realistic blurry/sharp video pairs. To the best of our knowledge, the proposed BSD is the first real-world video deblurring dataset.
- The experimental results demonstrate that our method achieves better deblurring performance both quantitatively and qualitatively than SoTA video deblurring methods with less computational cost.
- We have thoroughly done cross-validation between the proposed BSD and the synthetic datasets. The experimental results demonstrate that real-world dataset BSD outperforms the synthetic datasets in terms of generality of the trained models.

A short version of this paper was published in [53]. Compared to [53], there are two main extensions: (i) We upgraded the beam-splitter acquisition system with center-aligned scheme and collected a larger real-world dataset for experiments with more scenes, more motion patterns and more blur intensity settings; (ii) We conducted cross-validation experiments between our real-world dataset BSD and the synthetic datasets, including existing dataset synthesized from 240 fps videos and a self-made high-fps dataset synthesized from 2000 fps videos, to validate the effectiveness of BSD.

## 2 Related Works

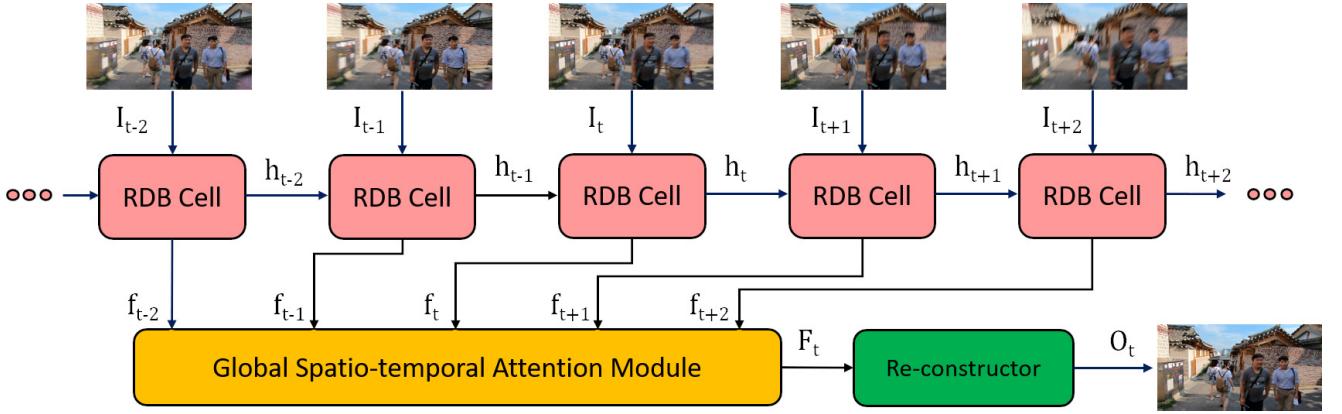
### 2.1 Video Deblurring

In recent years, video deblurring techniques become significant for daily life media editing and for advanced processing such as SLAM [20], 3D reconstruction [21] and visual tracking [45]. Research focus starts to shift from early single non-blind image deblurring [34, 40, 58] and single blind image deblurring [5, 8, 24, 30, 47] to the more challenging tasks such as blur decomposition [14, 35, 54], and video deblurring [27, 37, 42, 43, 55, 56].

Typically, the blur in a video has varying size and magnitude at different positions in each frame. In the

early work of video deblurring, [2] attempts to automatically segment moving blurred objects from the background and assumes a uniform blur model for them. Then, in view of different kinds of blur in different regions of an image, [46] tries to segment an image into two layers and generate segment-wise blur kernels for deblurring. More recently, there are some researches that estimate pixel-wise blur kernel with segmentation [32], or without segmentation [15]. However, these kernel based methods are quite expensive in computation and usually rely on human knowledge. An inaccurate blur kernel estimation will result in severe artifacts in the deblurred image. Alternatively, [6] uses homography as the underlying motion model. Instead of deconvolution with kernel estimation, it searches for the luckier pixels in adjacent frames and uses them to replace the less lucky pixels in the current frame. However, the homography assumption can only be applied to the blur caused by camera shake.

To overcome the above issues, researchers started to work on deep learning methods for video deblurring. CNN-based methods are used to handle the inter-frame relationship of video signal, such as [37], which makes the estimation of deblurred frame by using channel-concatenated neighboring frames with optional alignment using homography or optical flow. To better utilizing information from neighboring frames, EDVR [42] uses deformable convolutional operation [57] in the encoder stage. Recently, on the basis of alignment using off-the-shelf optical flow estimators [13, 38], CDVD-TSP [31] develops a temporal sharpness prior and a cascaded training approach to jointly optimize the network; while PVDNet [36] retrains a blur-invariant flow estimator and uses a pixel volume containing candidate sharp pixels to address motion estimation errors. However, additional flow estimator branch usually makes the model more computationally expensive. On the other hand, some researchers tend to focus on RNN-based methods because of their excellent performance for handling time-series signal. RNN-based methods can manage alignment implicitly through hidden states. For example, [44] employs RNN architecture to reuse the features extracted from the past frame, and [16] improves the deblurring performance by blending the hidden states in temporal domain. Then, [27] is proposed to iteratively update the hidden state via reusing RNN cell parameters and achieves impressive video deblurring performance while operating in real time. [56] proposes a unified RNN framework to generate spatially adaptive filters for alignment and deblurring. [43] introduces a novel framework that utilizes the motion magnitude prior as guidance for efficient deep video deblurring. In addition, [55] considers the problem of joint



**Fig. 2** Framework of the proposed efficient spatio-temporal recurrent neural network.  $I_t$  refers to the  $t^{th}$  input blurry frame;  $h_t$  and  $f_t$  refer to the extracted hidden state and hierarchical features of RDB-based RNN cell (see Sec. 3.2) from  $t^{th}$  frame;  $F_t$  refers to the fused features generated by GSA module (see Sec. 3.3) for  $t^{th}$  frame;  $O_t$  refers to the  $t^{th}$  deblurred frame by the proposed method.

rolling shutter correction and video deblurring in the case of using rolling shutter cameras.

In this paper, we adopt a RNN framework similar to [27]. Our method is different from [27] in that we integrate RDB into the RNN cell in order to exploit the potential of the RNN cell through feature reusing and generating hierarchical features for the current frame. Furthermore, we propose a GSA module to selectively merge effective hierarchical features from both past and future frames, which enables our model to utilize the spatio-temporal information more efficiently.

## 2.2 Deblurring Dataset

As data-driven methods become dominant in the field of deblurring techniques, high-quality deblurring datasets with training image/video pairs become increasingly important. Since obtaining ground truth for real blurry image/video has been a challenge for a while, researchers thereby turned to synthetic datasets to circumvent this problem. A common approach used to create blurry images is to convolve clean natural images with synthetic blur kernels, just as was done in these works [5, 19, 39, 48]. A more typical approach is to average consecutive sharp frames in a high-fps video to create visually realistic blur that forms due to relatively long exposure, such as DVD [37], GOPRO [26] and REDS [25]. Research on the synthesis of blur on RAW space [3] has also been carried out. However, the unnatural data distribution and artifacts in the synthetic deblurring datasets will inevitably affect the performance of the model in real-world situation.

Recently, researchers have resorted to designing customized hardware platforms to collect real-world dataset [4, 33, 50, 53, 55] for low-level vision tasks. Take

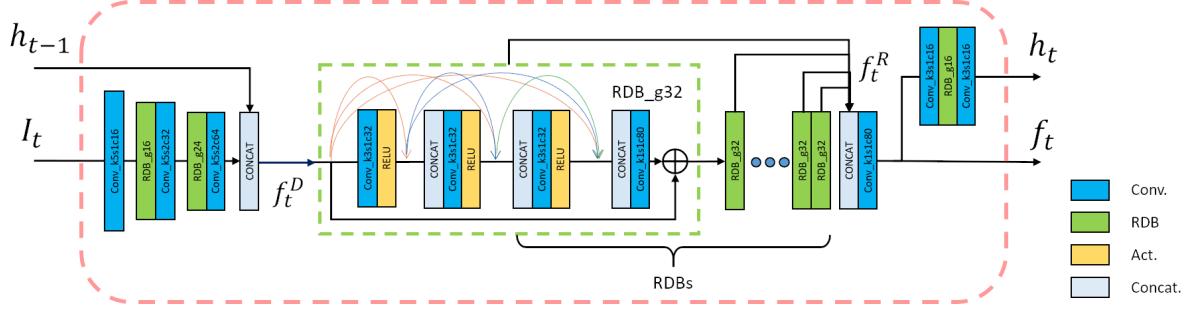
the image super-resolution task as an example, a zoom lens is used in [50] to collect high-resolution and low-resolution image pairs in static scenes. In this work, we design a beam-splitter acquisition system for the video deblurring task and contribute the first real-world video deblurring dataset to the community. We found another work [33] that proposed a similar system to ours in the same period for collecting deblurring dataset. However, unlike [33], we adopt the single object lens scheme to ensure precise alignment of the two cameras without receiving the effects of lens inter-reflection. Our carefully designed acquisition system is sufficient to capture blurry/sharp video pairs for both image and video deblurring, while the acquisition system in [33] can only be used to capture single blurry/sharp image pairs.

## 3 Proposed Method

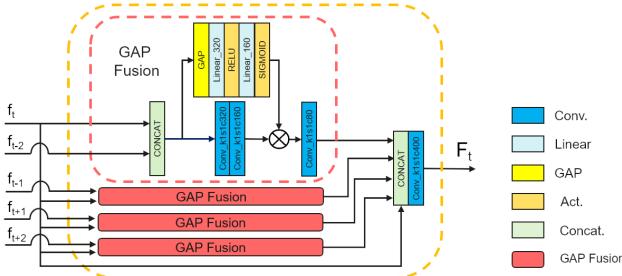
In this section, we will first give an overview of the proposed method in Sec. 3.1. Then we will go into details of RDB cell and GSA module in Sec. 3.2 and Sec. 3.3, respectively.

### 3.1 Overview

According to the characteristics of blur in the video, it may keep varying temporally and spatially, which makes deblurring problem intractable. In turn, it is possible that the blurred information in the current frame is relatively clear and complete in the past frames and future frames. When using RNN-based method to implement video deblurring, high-level features of the current frame will be generated to make the inference of deblurred image. Actually, some parts of the high-level



**Fig. 3** The structure of RDB-based RNN cell.  $h_t$  and  $h_{t-1}$  refer to the hidden state of past frame and current frame, respectively;  $I_t$  refers to the input blurry frame;  $f_t^D$  refers to the features after downsampling module;  $f_t^R$  refers to the hierarchical features generated by the RDB cell; As for the details of each layer and RDB module,  $k$ ,  $s$ ,  $c$  and  $g$  denote kernel size, stride, channels and growth rate, respectively.



**Fig. 4** The structure of global spatio-temporal attention module.  $f_{t-2}$ ,  $f_{t-1}$ ,  $f_{t+1}$ ,  $f_{t+2}$  and  $f_t$  refer to the hierarchical features of corresponding neighboring frames in the past and future and the current frame, respectively; *Linear* refers to fully convolutional layer; *GAP* refers to global average pooling fusion module;  $F_t$  refers to the output of GSA module, integrating the effective components of hierarchical features from neighboring frames generated by *GAP* fusion module.

features are worth saving and reusing for making up the loss information for other frames. Therefore, distributing part of computational resources to fuse informative features in past and future frames could be a method to effectively improve the efficiency of the neural network. Furthermore, how to improve RNN cell itself to extract high-level features with better spatial structure is critical to enhance the efficiency of the neural network. Starting from the above viewpoints, we integrate multiple residual dense blocks into RNN cell to generate hierarchical features and propose a global spatio-temporal attention module for effective feature fusion of neighboring frames.

The whole video deblurring process of our method is shown as Fig. 2. We denote the input frames of blurry video and corresponding output frames as  $\{I_t\}$  and  $\{O_t\}$  respectively, where  $t \in \{1 \dots T\}$ . Through RDB-based RNN cell, the model could get hierarchical features for each frame as  $\{f_t\}$ . To get the inference of latent frame  $O_t$ , the global spatio-temporal attention module takes current hierarchical feature  $f_t$  with two

past and two future features ( $f_{t-2}, f_{t-1}, f_{t+1}, f_{t+2}$ ) as input to perform feature fusion and generate  $F_t$  as output. Finally, through re-constructor module, the model can obtain the latent frame  $O_t$ .

### 3.2 RDB Cell: RDB-based RNN Cell

We adopt residual dense block [51, 52] into the RNN cell, named as RDB cell. The dense connections of RDB inherited from dense block (DB) [12] let each layer receive feature maps from all previous layers by concatenating them together in channel dimension. The output channels of each layer in RDB will keep the same size. This allows collective features to be reused and save the computational resources. Moreover, through local feature fusion, RDB could generate hierarchical features from convolutional layers in different depth with different size of receptive fields, which could provide better spatial information for image reconstruction.

The structure of RDB-based RNN cell is illustrated in Fig. 3. First, the current input frame  $I_t$  will be down-sampled and concatenated with last hidden state  $h_{t-1}$  to get shallow feature maps  $f_t^D$  as follows:

$$f_t^D = CAT(DS(I_t), h_{t-1}), \quad (1)$$

where  $CAT(\cdot)$  refers to concatenation operation;  $DS(\cdot)$  refers to down-sampling operation in the cell which consists of  $5 \times 5$  convolutional layers and RDB module. Then,  $f_t^D$  will be fed into a series of RDB modules. The collective outputs of stacked RDB modules are represented as  $f_t^R = \{f_t^{R_1}, \dots, f_t^{R_N}\}$ , where  $N$  refers to the number of RDB modules. RDB cell could obtain the global hierarchical features  $f_t$  by fusing the concatenation of local hierarchical features  $f_t^R$  with  $1 \times 1$  convolutional layer as follows:

$$f_t = Conv(CAT(f_t^R)), \quad (2)$$

where  $\text{Conv}(\cdot)$  refers to convolutional operation. Then, the hidden state  $h_t$  could be updated as follows:

$$h_t = H(f_t), \quad (3)$$

where  $H$  refers to the hidden state generation function, consisting of  $3 \times 3$  convolutional layer and RDB module. In short, while processing each frame in the video, the inputs of RDB cell are current blurry frame and previous hidden state. Then, RDB cell will generate the hierarchical features of this frame and update the hidden state as well.

### 3.3 GSA: Global Spatio-temporal Attention Module

The structure of GSA module is illustrated in Fig. 4. This module aims to extract and fuse the effective components of hierarchical features from future and past frames. Intuitively, the frames which are closer to the current frame in time domain are more likely to have useful information for deblurring of current frame. In the situation of real-time video deblurring, considering that the requirement of low computational cost for each output frame, the number of neighboring hierarchical features that will be fused into current frame should be limited. Furthermore, considering that delaying output by only several frames is usually acceptable, the hierarchical features from the future frames are available for the feature fusion. Therefore, the input to GSA will be hierarchical features of two frames before and after the current frame and the current frame itself as  $\{f_{t-2}, f_{t-1}, f_t, f_{t+1}, f_{t+2}\}$ . Inspired by Squeeze-and-Excitation (SE) block in [11], a submodule named global averaging pooling fusion is proposed, which takes features of current frame and a neighboring frame as input to filter out effective hierarchical features  $f_{t+i}^e$  from the neighboring frame as follows:

$$f_{t+i}^c = \text{CAT}(f_t, f_{t+i}), \quad (4)$$

$$f_{t+i}^e = L(\text{GAP}(f_{t+i}^c)) \otimes P(f_{t+i}^c), \quad (5)$$

where  $i \in \{-2, -1, 1, 2\}$ ;  $\text{GAP}(\cdot)$  refers to global averaging pooling [23];  $L(\cdot)$  refers to a series of linear transformation with activation function as  $\text{ReLU}$  [28] and  $\text{Sigmoid}$  function for channel weight generation;  $P(\cdot)$  refers to a series of  $1 \times 1$  convolutional operations for feature fusion. Finally, GSA module will fuse the  $f_t$  with all effective hierarchical features from neighboring frames to get the output  $F_t$  as follows:

$$F_t = \text{Conv}(\text{CAT}(f_{t-2}^e, f_{t-1}^e, f_{t+1}^e, f_{t+2}^e, f_t)). \quad (6)$$

The output  $F_t$  of GSA module will be upsampled by deconvolutional layers [7] in re-constructor module for generating latent image for the current frame.

## 4 Dataset for Video Deblurring

In this section, we first briefly introduce the mainstream synthetic video deblurring datasets and the corresponding simulation method. Then we present the details of our beam-splitter acquisition system and the corresponding BSD dataset.

### 4.1 Synthesized Dataset

At present, there are still very limited methods for building a video deblurring dataset. The mainstream way is to average a series of consecutive short-exposure images in order to mimic the phenomenon of blur caused by relatively long exposure time [17]. The process of that can be described as follows:

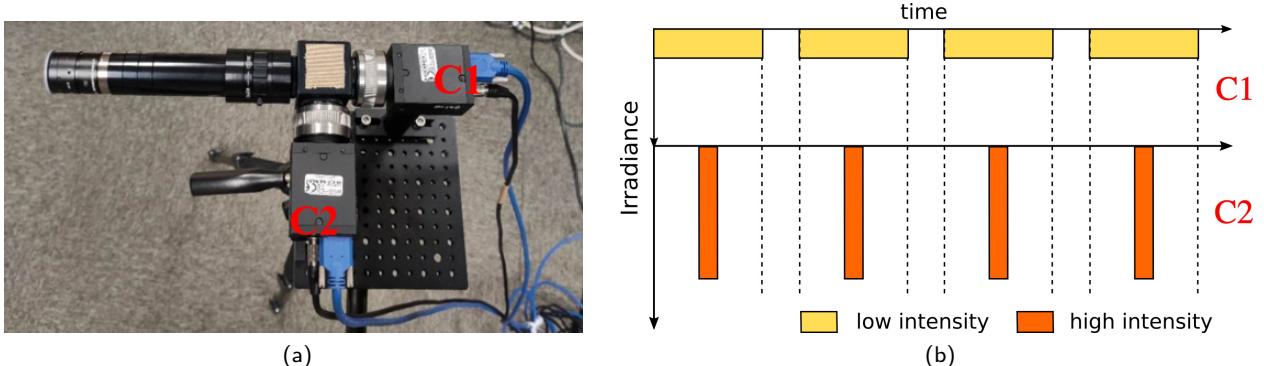
$$I_b = \text{CRF} \left( \frac{1}{N} \sum_{n=1}^N S^n \right), \quad (7)$$

where  $N$ ,  $S^n$  denote the number of sampled high-FPS frames and the signal of  $n^{th}$  sharp frame;  $\text{CRF}$  denotes the camera response function.

This kind of method requires a high-speed camera to capture high-fps video and then synthesizes pairs of sharp and blurry videos based on the high-FPS videos. Before averaging, video frame interpolation algorithms [29] are usually used in advance to supplement the information of missing exposure time in the high-fps video, which helps to avoid unnatural spikes or steps in the blur trajectory caused by inadequate frame rate [44]. Mainstream public datasets for video deblurring, such as GOPRO [26] and REDS [25], are all born by the above method. There are 22 training video sequences and 11 evaluation video sequences in GOPRO with 2103 training samples and 1111 evaluation samples respectively. As for REDS, there are 240 training sequences and 30 evaluation sequences with 100 frames for each sequence.

**Table 1** Configuration of the proposed BSD dataset

	train	validation	test
sequences	60	20	20
frames/seq.	100	100	150
frames	6000	2000	3000
resolution		$640 \times 480$	
frequency		15 fps	
settings	1ms-8ms, 2ms-16ms, 3ms-24ms		



**Fig. 5** A beam splitter acquisition system for building video deblurring dataset. (a) is the profile of our beam splitter acquisition system.  $C_1$  and  $C_2$  refer to two same cameras with different configurations for generating blurry and sharp videos, respectively; (b) shows the center-aligned exposure scheme of  $C_1$  and  $C_2$  to generate blurry/sharp video pairs.



**Fig. 6** Samples of blurry/sharp image pairs in BSD. (a) represents the case where there is only camera ego-motion. (b) represents the case where only the object is moving. (c) represents the case where the object and the camera are moving in the opposite directions. (d) represents the case where the object and the camera are moving in the same direction.

#### 4.2 Beam-Splitter Deblurring Dataset (BSD)

It is questionable whether aforementioned synthetic way truly reflects the blur in real scenarios. Here, we provide a new solution for building video deblurring dataset by using a beam splitter acquisition system with two synchronized cameras, as shown in Fig. 5(a). In our solution, by controlling the length of exposure time and strength of exposure intensity during video shooting as shown in Fig. 5(b), the system could obtain a pair of sharp and blurry videos in one shot. We adopt center-aligned synchronization scheme to properly delay the pulse of  $C_2$ , so that the sharp exposure time lies exactly in the middle of the blurry exposure time. Compared to start-aligned or end-aligned synchronization scheme, center-aligned scheme can avoid

large displacement between blurry/sharp image pairs. To further realize photometric alignment, we insert a 12.5% neutral density filter in the front of  $C_1$  to reduce the irradiance intensity as 1/8 of camera  $C_2$ . Correspondingly, the exposure time of camera  $C_1$  is always kept as 8 times of camera  $C_2$ .

The configurations of the proposed BSD dataset is illustrated in Table 1. We collected blurry/sharp video sequences for three different blur intensity settings (sharp exposure time – blurry exposure time), 1ms–8ms, 2ms–16ms and 3ms–24ms, respectively. The acquisition frequency is 15fps. For each setting, the training and validation sets have 60 and 20 video sequences with 100 frames in each, respectively, and the test set has 20 video sequences with 150 frames in each. There are 11000 blurry/sharp image pairs in total for

each setting. The resolution of all videos is uniformly  $640 \times 480$ . Blurry/sharp image pairs of different motion patterns are illustrated in Fig. 6. The 1<sup>st</sup> row shows blurry images, and the 2<sup>nd</sup> row shows the corresponding sharp images. Specifically, Fig. 6(a) represents the case where only the camera is moving; Fig. 6(b) represents the case where only the object is moving while the camera is static. Thus, the background is sharp but the car is blurry; Fig. 6(c) represents the case where the camera and the object are moving in the opposite directions; Fig. 6(d) represents the case where the camera and the object are moving in the same direction. As a result, the car appears sharp while the background is blurry.

## 5 Experiments and Results

### 5.1 Implementation Details

For comparison on synthetic datasets, we conduct the experiments on GOPRO and REDS, respectively. As for GOPRO, we choose the same version as [27]. Due to the huge size of REDS dataset and limited computational resources, we train our model and other SoTA models only on first-half training sequences of REDS (120 sequences) for comparison. We train the model for 500 epochs by ADAM optimizer [18] ( $\beta_1 = 0.9$ ,  $\beta_2 = 0.999$ ) with initial learning rate as  $10^{-4}$  (decay rate as 0.5, decay step as 200 epochs). We use RGB patches of  $256 \times 256$  size in subsequence of 10 frames as input to train the models. Also, we implement horizontal and vertical flipping for each subsequence as data augmentation. Mini-batch size is set to 4. The loss function is defined as MSE loss  $\mathcal{L}_{MSE}$  as follows:

$$\mathcal{L}_{MSE} = \frac{1}{TCHW} \sum_{t=1}^T \|O_t - O_t^{GT}\|_2^2, \quad (8)$$

where  $T$ ,  $C$ ,  $H$ ,  $W$  denote the number of frames and the number of channel, height, width for each frame;  $O_t^{GT}$  refers to the ground truth of the  $t^{th}$  frame.

Whereas, in the experiments on the proposed real-world dataset BSD, we adopt cosine annealing schedule to adjust learning rate with default learning rate as  $3 \times 10^{-4}$ . Mini-batch size is set to 8. Length of subsequence is 8. We use charbonnier loss function  $\mathcal{L}_{char}$  ( $\epsilon = 1 \times 10^{-3}$ ) as follows:

$$\mathcal{L}_{char} = \frac{1}{TCHW} \sum_{t=1}^T \left\| \sqrt{(O_t - O_t^{GT})^2 + \epsilon^2} \right\|. \quad (9)$$

**Table 2** Quantitative results on both GOPRO and REDS datasets. Cost refers to the computational cost of the model for deblurring one frame of HD (720P) video in terms of GMACs. The meaning of cost is same for other tables and figures in this paper. For our model,  $B_{\#}$  and  $C_{\#}$  denote the number of RDB blocks in RDB cell and the number of channels for each RDB block, respectively.

Model	GOPRO		REDS		Cost
	PSNR	SSIM	PSNR	SSIM	
STRCNN [16]	28.74	0.8465	30.23	0.8708	276.20
DBN [37]	29.91	0.8823	31.55	0.8960	784.75
IFI-RNN ( $c2h1$ ) [27]	29.79	0.8817	31.29	0.8913	116.29
IFI-RNN ( $c2h2$ ) [27]	29.92	0.8838	31.35	0.8929	167.09
IFI-RNN ( $c2h3$ ) [27]	29.97	0.8859	31.36	0.8942	217.89
STFAN [56]	30.51	0.9054	32.03	0.9024	566.61
CDVD-TSP [31]	30.94	0.9153	32.57	0.9161	5211.28
PVDNet [36]	32.13	0.9322	33.92	0.9322	1754.90
ESTRNN ( $B_9C_{60}$ )	30.12	0.8837	31.64	0.8930	92.57
ESTRNN ( $B_9C_{65}$ )	30.30	0.8892	31.63	0.8965	108.20
ESTRNN ( $B_9C_{70}$ )	30.45	0.8909	31.94	0.8968	125.55
ESTRNN ( $B_9C_{75}$ )	30.58	0.8923	32.06	0.9022	143.71
ESTRNN ( $B_9C_{80}$ )	30.79	0.9016	32.33	0.9060	163.61
ESTRNN ( $B_9C_{85}$ )	31.01	0.9013	32.34	0.9074	184.25
ESTRNN ( $B_9C_{90}$ )	31.07	0.9023	32.63	0.9110	206.70

### 5.2 Experiments on Synthetic Datasets

For fair comparison, we mainly compare our method with lightweight deblurring methods without using pretrained optical flow estimator. We note that the SoTA models, such as CDVD-TSP and PVDNet, use a pre-trained optical flow estimator (PWC-Net [38] for CDVD-TSP and LiteFlowNet [13] for PVDNet) as part of the model, which means additional information is introduced into the model. Yet we also present the comparison results with CDVD-TSP [31] and PVDNet [36] for your reference.

#### 5.2.1 Comparison on GOPRO

First, we compare our method with the SoTA video deblurring methods on GOPRO dataset. We implement 7 variants of our model with different computational cost by modifying the number of channels ( $C_{\#}$ ) of the model and keeping the number of RDB blocks ( $B_{\#}$ ) as 9. The larger  $C_{\#}$  is, the higher computational cost it needs. We report the deblurring performance and the corresponding computational cost for processing one frame in the video of all compared models in terms of PSNR [10], SSIM and GMACs, respectively, in Table 2. From the perspective of quantitative analysis, it is clear that our model can achieve higher PSNR and SSIM value with less computational cost, which means our model has higher network efficiency. To further validate the deblurring performance of proposed model, we also show the deblurred image generated by each model, as illustrated in Fig. 7. We can see that the proposed model can restore sharper image with more details, such as



**Fig. 7** Visual comparisons on GOPRO [26].

the textures of tiles on the path and the characters on the poster.

### 5.2.2 Comparison on REDS

We also do the comparison on REDS, which has more diverse scenes from different places. From Table 2, we can see our model  $B_9C_{90}$  achieves best results as 32.63 PSNR with only around 200 GMACs computational cost for one 720P frame. Even our small model  $B_9C_{60}$  with cost less than 100 GMACs can achieve same level performance as  $c2h3$  of IFI-RNN, the computational cost of which is as twice as the former. In terms of qualitative results illustrated in Fig. 8, the proposed model can significantly reduce ambiguous parts for the deblurred frame, and the restored details such as the texture of the wall, characters, and human body are closer to the ground truth.

### 5.2.3 Network Efficiency Analysis

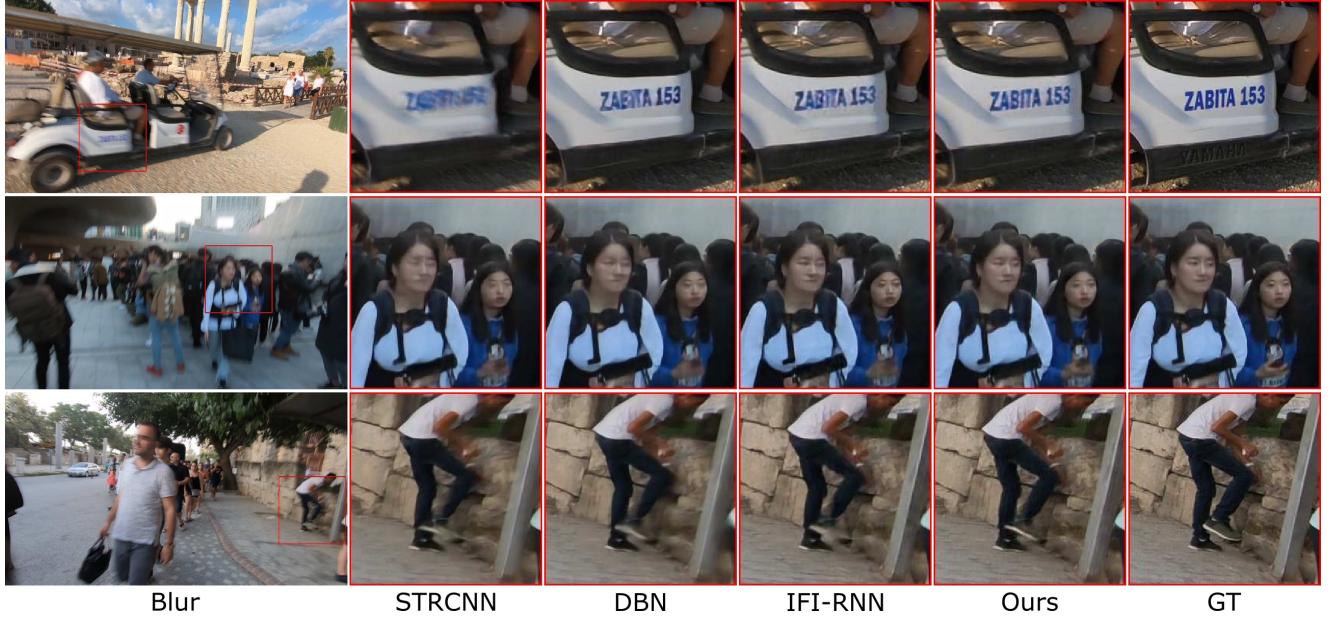
We collect the computational cost for one frame as well as the performance (PSNR) of the SoTA lightweight image and video deblurring models on GOPRO dataset, as illustrated in Fig. 1(b). The proposed model includes 7 red nodes that represent different variants of our ESTRNN from  $B_9C_{60}$  to  $B_9C_{90}$  in Table 2. Also, the three blue nodes represent different variants of IFI-RNN as  $c2h1$ ,  $c2h2$  and  $c2h3$ . Because the computational cost of different models varies drastically, we take  $\log_{10}(\text{GMACs})$  as abscissa unit to better display the results. An ideal model with high network efficiency will

locate at upper-left corner of the coordinate. The proposed models are closer to the upper-left corner than the existing image or video deblurring models, which reflects the high network efficiency of our model.

We further present computational cost and inference time comparison in Table 3. Because the parallelism of RNNs is in general worse than CNNs, RNN-based models do not have a considerable advantage in inference time, although they have a greater advantage in GMACs. However, our model still demonstrates efficiency improvements. Take the small variant ESTRNN ( $B_9C_{60}$ ) as example, it is superior to other models in terms of performance, inference speed and computational cost. We also note that the parallelism of RNNs can be improved according to related literature [22, 49].

### 5.2.4 Ablation Study

We conduct an ablation study to demonstrate the effectiveness of the high-level feature fusion strategy, RDB cell, as well as GSA module, as shown in Table 4. When ablating the modules, we keep the computational cost almost unchanged by adjusting the number of channels ( $C_\#$ ) for fair comparison. Specifically, without using fusion strategy means that the model directly reconstructs the result according to high-level features only from current frame; without RDB cell, the model will use residual block [9] instead, in the same way as [26] does; without GSA module, high-level features will be directly concatenated in channel dimension. The results clearly demonstrate that each module or design can improve the deblurring efficiency, because each mod-



**Fig. 8** Visual comparisons on REDS [25].

**Table 3** Computational cost and inference time comparison. We show the performance of each model in terms of GMACs, Million parameters, inference time (ms), frame rate (fps), and the corresponding PSNR score. The size of test image is  $1280 \times 720$ , and the test hardware is GeForce RTX 2080 Ti. The average value is reported.

	STRCNN [16]	DBN [37]	IFI-RNN [27]	ESTRNN (B9C60)	ESTRNN (B15C80)
GMACs	276.20	784.75	217.89	92.57	203.74
M Parameters	0.93	15.31	1.64	0.99	2.47
Inference Time (ms)	42.03	61.9	67.0	33.7	62.2
Frame rate (fps)	23.79	16.16	14.93	29.66	16.07
PSNR	28.74	29.91	29.97	30.12	31.27

**Table 4** Ablation study of ESTRNN. Fusion refers to the fusion strategy that utilizes the high level features from neighboring frames. The experiments are conducted on GOPRO.

Model	Fusion	RDB Cell	GSA	PSNR	Cost
$B_9C_{110}$	×	×	×	30.29	163.48
$B_9C_{100}$	✓	×	×	30.46	165.59
$B_9C_{100}$	×	✓	×	30.51	168.56
$B_9C_{90}$	✓	✓	×	30.55	161.28
$B_9C_{85}$	×	✓	✓	30.69	162.69
$B_9C_{80}$	✓	✓	✓	30.79	163.61

**Table 5** Effectiveness of number of RDB blocks. The experiments are conducted on GOPRO.

	$B_3C_{80}$	$B_6C_{80}$	$B_9C_{80}$	$B_{12}C_{80}$	$B_{15}C_{80}$
PSNR	29.74	30.31	30.79	31.03	31.27
Cost	123.03	143.32	163.31	183.90	204.19

ule can improve the overall performance of model when the computational cost keeps unchanged. Besides, we show visual results to provide some intuitions about GSA module for high-level feature fusion in Fig. 9. In these two cases, the results of model without using GSA

**Table 6** Effectiveness of number of neighboring frames used by GSA module.  $F_{\#}$  and  $P_{\#}$  refers to the number of future and past frames used by the model. The base model is  $B_9C_{80}$ . The experiments are conducted on GOPRO.

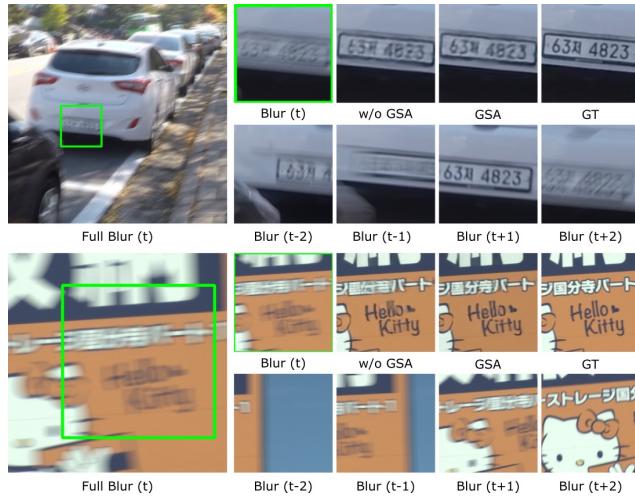
	$F_0P_1$	$F_0P_2$	$F_0P_3$	$F_1P_1$	$F_2P_2$	$F_3P_3$
PSNR	30.54	30.57	30.69	30.58	30.79	30.82
Cost	119.93	133.75	148.31	133.75	163.61	196.42

to fuse features from neighboring frames (only features from current frame as input for the re-constructor) are inferior to the results of the complete model. The gain should come from the time instances, such as  $t+1$  in the first case and  $t+2$  in the second case, where the “blurry” input image has relatively sharp but misaligned appearance.

We further explore the effectiveness of the number of RDB blocks and the number of past and future frames used by the model as Table 5 and Table 6, respectively. First, from the perspective of the number of RDB blocks, this is intuitive that more blocks which means more computational cost will achieve better per-

**Table 7** Quantitative results on BSD dataset. The configuration of ESTRNN is  $B15C80$ . The configuration of IFI-RNN is  $C2H3$ .

Model	1ms–8ms		2ms–16ms		3ms–24ms	
	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
STRCNN [16]	32.20	0.924	30.33	0.902	29.42	0.893
DBN [37]	33.22	0.935	31.75	0.922	31.21	0.922
SRN [41]	31.84	0.917	29.95	0.891	28.92	0.882
IFI-RNN [27]	33.00	0.933	31.53	0.919	30.89	0.917
STFAN [56]	32.78	0.922	32.19	0.919	29.47	0.872
CDVD-TSP [31]	33.54	0.942	32.16	0.926	31.58	0.926
PVDNet [36]	33.34	0.937	32.22	0.926	31.35	0.923
ESTRNN	33.36	0.937	31.95	0.925	31.39	0.926



**Fig. 9** Visual ablation study of the GSA module.

formance. If we compare the variant  $B15C80$  with variant  $B9C90$  in Table 2 which has almost same computational cost, we can find that it is better to increase the number of RDB blocks rather than the channels, when the number of channels is relatively sufficient. As for the number of neighboring frames, Table 6 shows that, considering the increased computational cost, the benefit of using more neighboring frames as  $F_3P_3$  is relatively small. Besides, the results of  $F_0P_1$ ,  $F_0P_2$  and  $F_0P_3$  show that the proposed model can still achieve comparative good results even without high-level features from the future frames.

### 5.3 Experiments on Real-world Dataset BSD

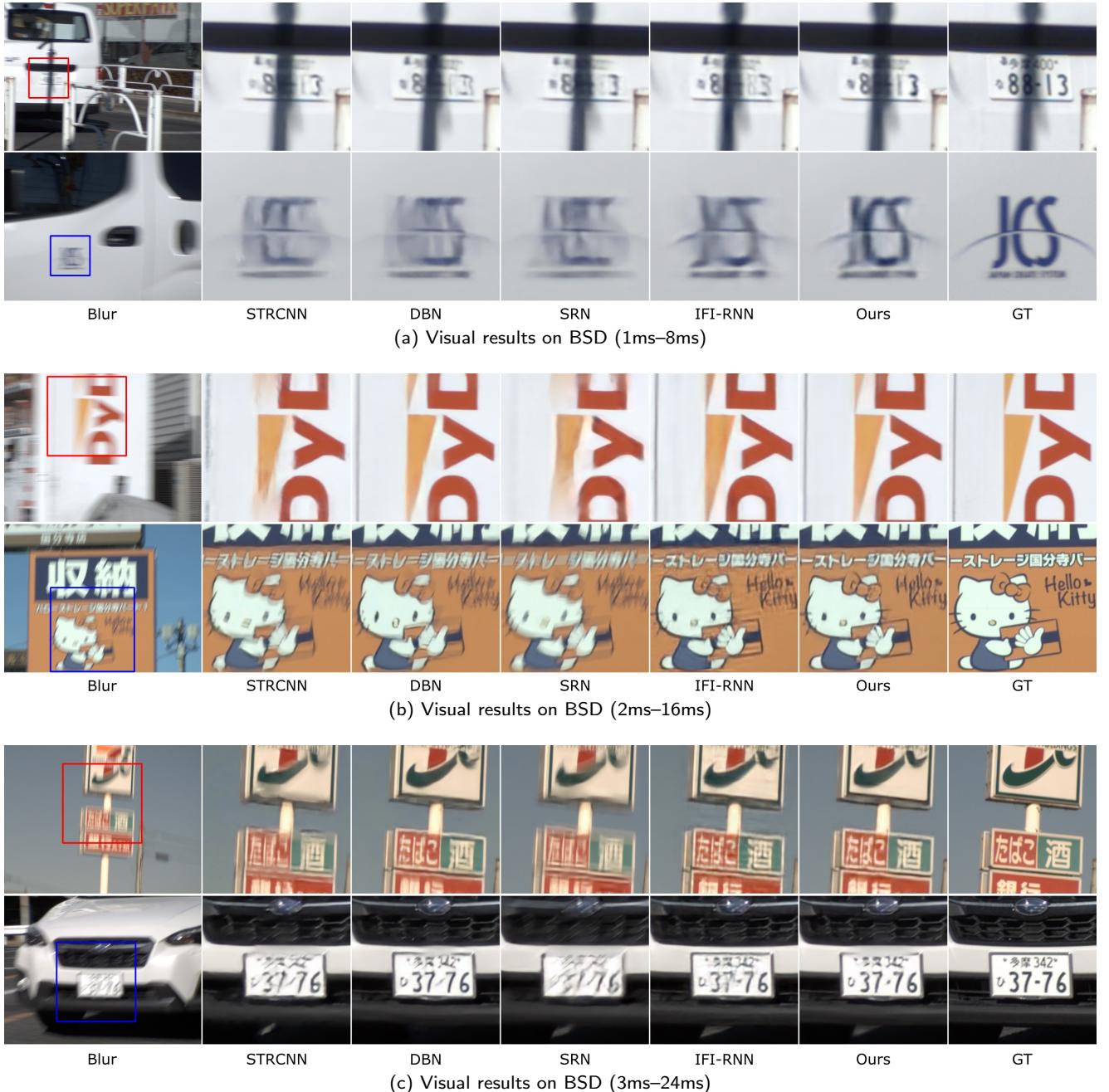
We also conduct comparison experiments on the proposed BSD dataset with different blur intensity settings. The visual results for 1ms–8ms, 2ms–16ms, and 3ms–24ms are illustrated in Fig. 10(a), Fig. 10(b) and Fig. 10(c), respectively. The proposed method ( $B15C80$ ) achieves more visually appealing results in all settings of the real-world deblurring dataset. The

quantitative results are shown as Table 7. It indicates that the dataset setting with longer exposure time, *i.e.*, higher blur intensity, is more difficult to restore. Our method achieves the best PSNR and SSIM scores of 33.36dB and 0.937 for 1ms–8ms setting, while the best scores of 31.39dB and 0.296 for 3ms–24ms setting. Both qualitative and quantitative results verify the effectiveness of our method on real-world video deblurring task. Also, comparison results with huge models with pre-trained optical flow estimator are presented in the supplemental materials.

## 5.4 Dataset Cross-validation

### 5.4.1 Qualitative cross-validation with synthetic datasets

To verify the advantages of using real-world data for training, we further conduct cross-validation between BSD and GOPRO. The predicted results for GOPRO by using our model ( $B15C80$ ) trained on BSD (2ms–16ms) are illustrated in Fig. 11(a); while the predicted results for BSD (2ms–16ms) by using our model trained on GOPRO are illustrated in Fig. 11(b). The comparison in Fig. 11 demonstrates that the model trained on real-world dataset has much better generalization ability than the model trained on synthetic dataset. The model trained on BSD has decent deblurring performance on GOPRO. In contrast, the model trained on GOPRO cannot deblur well on BSD but introduce severe artifacts. The above observations also apply to the other two BSD settings, 1ms–8ms and 3ms–24ms. We also show the results of cross-validation between BSD and REDS [25] in Fig. 12, as well as BSD and DVD [37] in Fig. 13. The results demonstrate that REDS dataset with higher quality images also suffers from the same issue. While the model trained with DVD performed better than the model trained with the GOPRO dataset or the REDS dataset when tested on real-world data. However, even if DVDs perform better than other syn-



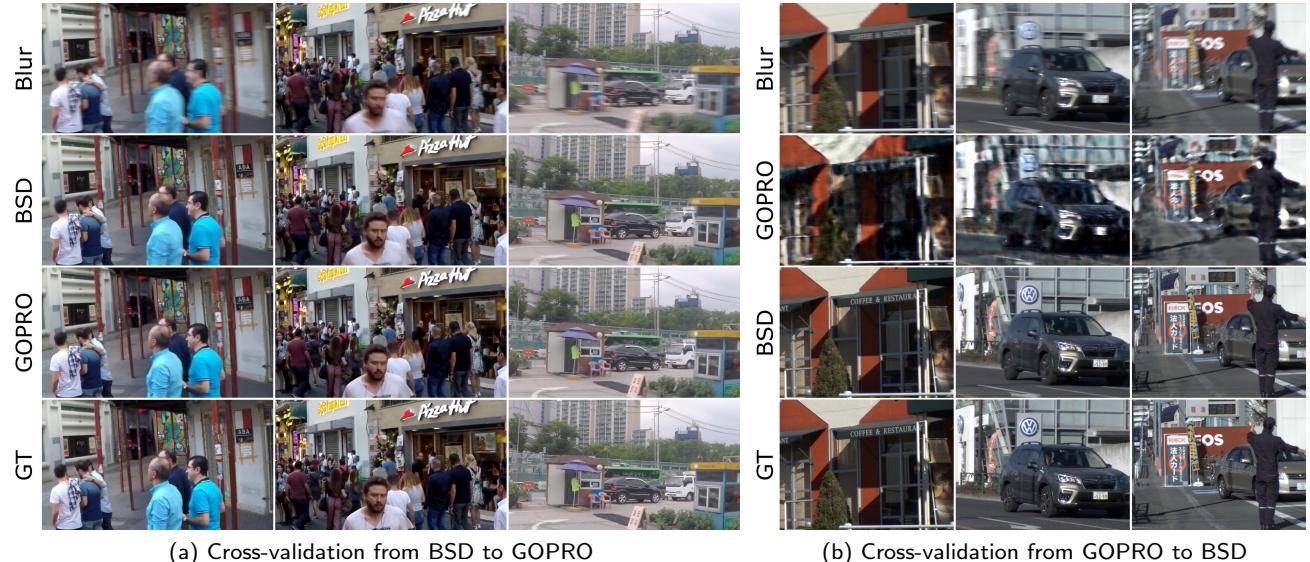
**Fig. 10** Visual comparisons on different settings of the proposed BSD.

thetic datasets, the trained model still produces undesired artifacts when processing real-world images with certain strongly blurred regions. This validates that real datasets are still the preferred choice for training deblurring models.

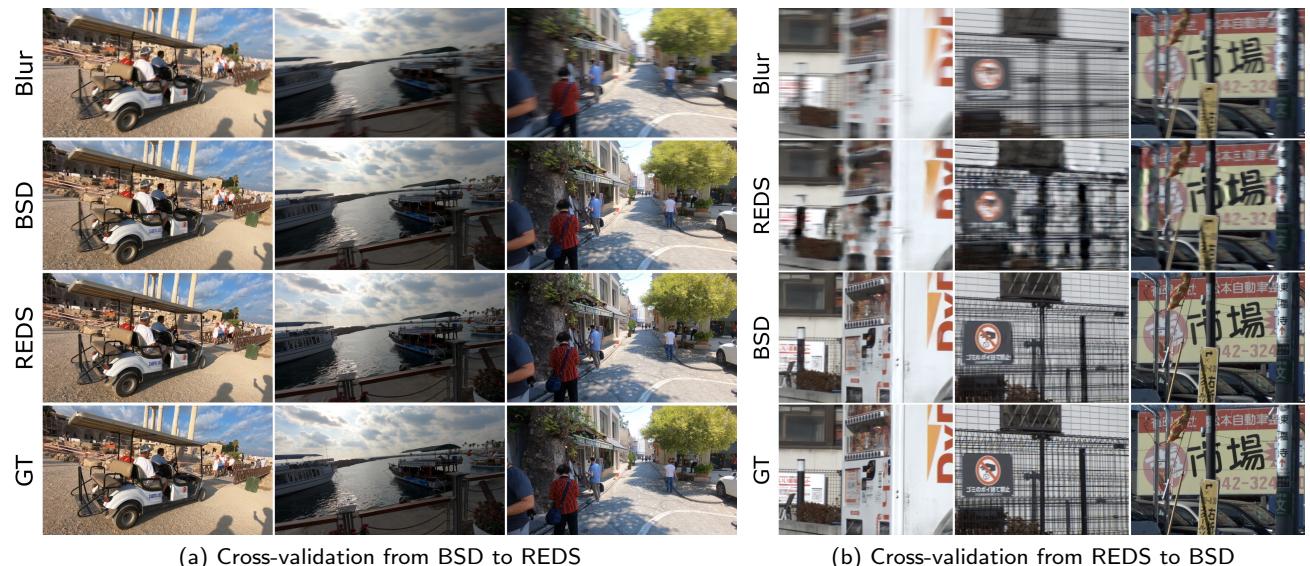
#### 5.4.2 Quantitative cross-validation with synthetic datasets

We also implement the quantitatively comparison for dataset cross-validation, as illustrated in Table. 8. We

show the bottom-score for each dataset in the first row. This bottom-score is calculated by using the original blur input without any processing and the corresponding sharp ground-truth. If the scores calculated from the processed images of the model are lower than the bottom-score, it can be considered that the model has a negative impact on the test data set in general. We can find that the model trained using the BSD can consistently obtain at least positive gains on the synthetic datasets. However, the model trained using the



**Fig. 11** Cross-validation between BSD and GOPRO.



**Fig. 12** Cross-validation between BSD and REDS.

GOPRO and REDS will result in a lower performance than the bottom-score of the BSD dataset. The model trained on DVD can also achieve positive gain on BSD, which is consistent with the observation of qualitative results.

#### 5.4.3 Possible factors affecting the generalization ability of synthetic data

The key difference between synthetic data pairs and real-world data pairs is that synthetic data is averaging with discrete images to simulate the formation of blur. We believe that the continuity of the discrete im-

ages used for synthesis and the noise distribution of the synthetic blur are two important factors that affect the generalization ability of the synthetic dataset.

First, we assume one of the reasons for causing the unsatisfactory results of the model trained on synthetic dataset is the insufficient frame rate of the original captured high-fps video (240 fps for GOPRO and 120 fps for REDS). Although video frame interpolation algorithms can be used to increase the frame rate, it is questionable whether the interpolated frames match the distribution of natural images.

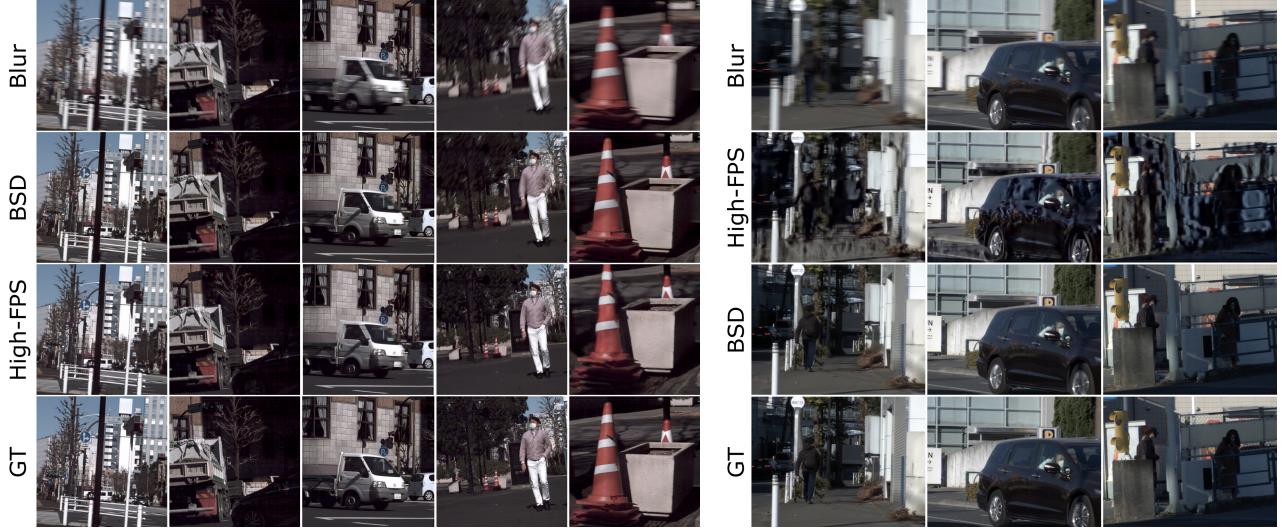


(a) Cross-validation from BSD to DVD

(b) Cross-validation from DVD to BSD

**Fig. 13** Cross-validation between BSD and DVD.**Table 8** Dataset cross-validation in terms of PSNR/SSIM. We choose the 2ms16ms setting for the BSD dataset. The setting of ESTRNN is (B15C80).

	BSD	GOPRO	REDS	DVD
Blur	26.64/0.818	25.47/0.785	26.28/0.769	27.23/0.813
BSD	31.95/0.925	26.46/0.817	27.00/0.801	27.88/0.844
GOPRO	19.48/0.598	31.27/0.903	28.21/0.829	28.90/0.869
REDS	24.14/0.773	28.43/0.869	32.82/0.915	28.06/0.848
DVD	28.67/0.875	26.57/0.820	26.64/0.787	30.68/0.897



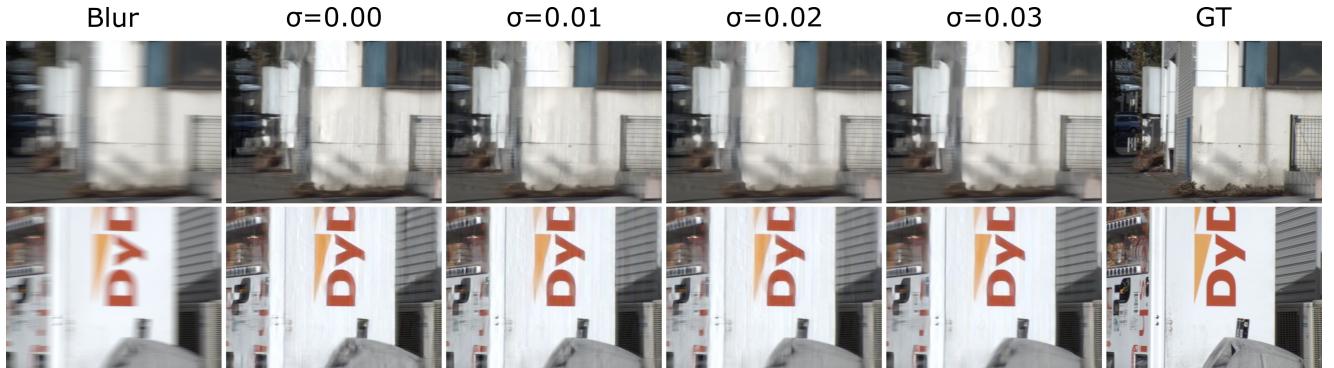
(a) Cross-validation from BSD to high-fps synthetic dataset

(b) Cross-validation from high-fps synthetic dataset to BSD

**Fig. 14** Cross-validation between BSD and high-fps synthetic dataset.

To figure out how insufficient frame rate affects the quality of the synthetic deblurring dataset, we use a

camera with a much higher frame rate (2000 fps) to record videos for making a new synthetic dataset. In



**Fig. 15** BSD test results of models (ESTRNN B15C80) trained by using DVD with additional Gaussian noise.

this case, the readout time is negligible and the exposure time is very close to 0.5 ms, thus video interpolation is not needed to supplement the missing information. We build a high-fps synthetic dataset with the same setting as BSD (2ms–16ms, 15fps) using the captured 2000 fps videos. Then, we conduct another cross-validation experiment between BSD (2ms–16ms) and the synthetic high-fps dataset to investigate whether a high enough frame rate could solve the problem of poor migration of the synthetic dataset. The results are illustrated in Fig. 14. Basically, the experimental results are consistent with the cross-validation between BSD and GOPRO, i.e., the model trained on BSD can work well on the synthetic high-fps dataset (see Fig. 14(a)), but not vice-versa (see Fig. 14(b)). Therefore, simply satisfying sufficient continuity cannot improve the generalization ability of the synthetic data.

From the perspective of noise distribution of blurred images, the present synthesis method cannot simulate the noise generated in real blurred images at the RAW acquisition and ISP stages. The process of averaging successive sharp images suppresses some of the noise. Thus, a likely reason for the one-sided relationship is that the model trained on BSD has seen more complex noise models so it can handle synthetic data with noise suppressed, which is the simpler case, while the reverse is difficult.

In addition, we trained a series of models on DVD by adding Gaussian noise to the RGB space of blurred images with different standard deviations  $\sigma$ . The visual results in Fig. 15 indicate that adding the appropriate noise can slightly improve artifact, but it is still far from eradicating it. Starting from RAW space to synthesize blur and considering the appropriate noise model should be a promising direction.

The above experiments demonstrate the significance of the real-world dataset and indicate that it is not trivial to produce a high-quality deblurring datasets by synthetic methods.

### 5.5 Test on Third-part Real-world Videos

Our real-world dataset BSD is obtained under some specific exposure settings. To verify the generalization ability of BSD, we further verify our model trained by using BSD on the commonly used real-world videos from DVD [6, 37], as illustrated in Fig. 16. We can see that our model trained on the synthetic datasets, GOPRO [26] and REDS [25] inevitably introduce undesired artifacts to the final results, such as the cover of the book, the wheel of the car, and the body of the man. Our model trained on BSD (2ms16ms) works well in most of these cases. It is worth noting that in the bicycle example, even though the model trained on BSD cannot recover the very challenging blurred bicycle wheel, the model does not force to add some artifacts. We believe that this is advantageous in terms of visual perception and can be further addressed by expanding the dataset.

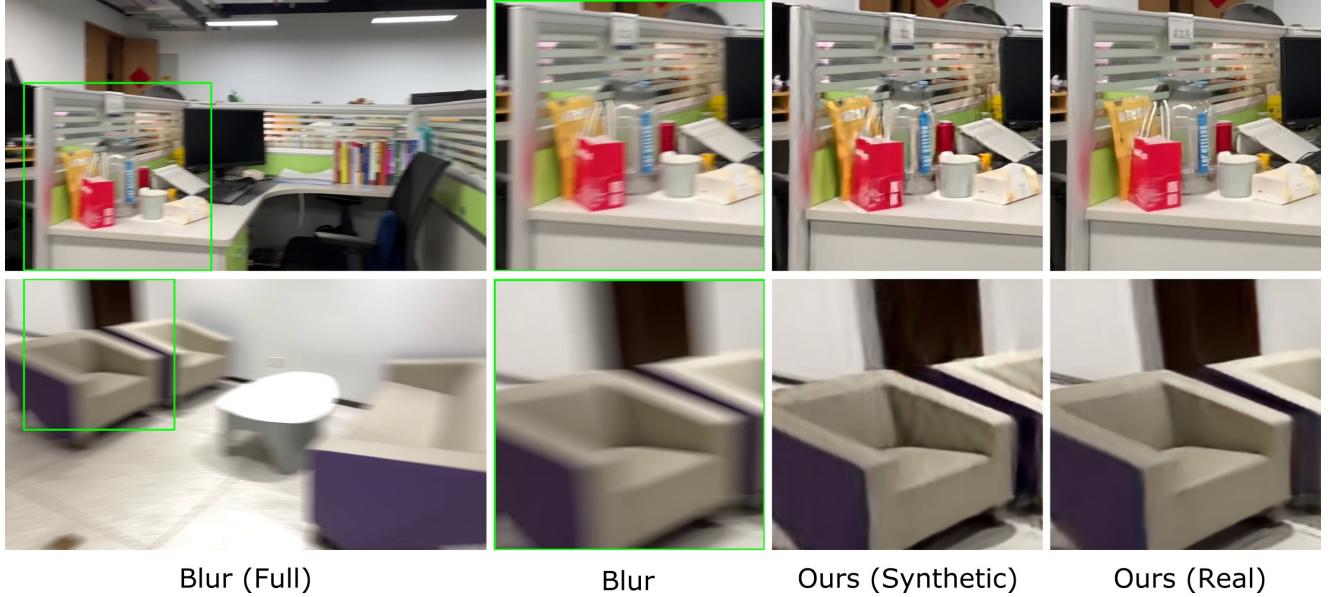
We also test our model by shooting blurred videos without any specific constraints using iPhone 13. The predicted results from our model trained on synthetic dataset GOPRO and on our real-world dataset BSD are illustrated in Fig. 17. It is clear that the model trained on real-world data successfully restored the latent images and generated clearer details such as the handle of the bag. While the model trained on synthetic dataset suffered from unsharp details and undesired artifacts.

## 6 Conclusion

In this paper, we proposed a novel RNN-based method for more computationally efficient video deblurring. Residual dense blocks were adopted to the RNN cell to generate hierarchical features from current frame for better restoration. Moreover, to make full use of the spatio-temporal correlation, our model utilized the global spatio-temporal fusion module for fusing the effective components of hierarchical features from past and future frames. Furthermore, we have developed



**Fig. 16** Qualitative results on real-world videos from [6, 37]



**Fig. 17** Testing on real-world blurry videos from iPhone 13. Ours (Synthetic) denotes the results of ESTRNN trained on synthetic dataset GOPRO. Ours (real) denotes the results of ESTRNN trained on real-world dataset BSD (2ms-16ms).

a beam-splitter acquisition system and contributed the first real-world dataset for image/video deblurring tasks. The experimental results show that our model is more computationally efficient for video deblurring, which can achieve better performance with less computational cost. Cross-validation experiments between real-world and synthetic datasets demonstrate the high generality of the proposed BSD dataset.

**Acknowledgements** This work was supported in part by JSPS KAKENHI Grant Numbers JP20H05951 and JP20H05953.

## References

- Bahdanau, D., Cho, K., Bengio, Y.: Neural machine translation by jointly learning to align and translate.

- arXiv preprint arXiv:1409.0473 (2014)
2. Bar, L., Berkels, B., Rumpf, M., Sapiro, G.: A variational framework for simultaneous motion estimation and restoration of motion-blurred video. In: 2007 IEEE 11th International Conference on Computer Vision, pp. 1–8. IEEE (2007)
  3. Cao, M., Zhong, Z., Fan, Y., Wang, J., Zhang, Y., Wang, J., Yang, Y., Zheng, Y.: Towards real-world video deblurring by exploring blur formation process. arXiv preprint arXiv:2208.13184 (2022)
  4. Cao, M., Zhong, Z., Wang, J., Zheng, Y., Yang, Y.: Learning adaptive warping for real-world rolling shutter correction. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 17785–17793 (2022)
  5. Chakrabarti, A.: A neural approach to blind motion deblurring. In: European conference on computer vision, pp. 221–235. Springer (2016)
  6. Cho, S., Wang, J., Lee, S.: Video deblurring for hand-held cameras using patch-based synthesis. ACM Transactions on Graphics (TOG) **31**(4), 1–9 (2012)
  7. Dumoulin, V., Visin, F.: A guide to convolution arithmetic for deep learning. ArXiv e-prints (2016)
  8. Goldstein, A., Fattal, R.: Blur-kernel estimation from spectral irregularities. In: European Conference on Computer Vision, pp. 622–635. Springer (2012)
  9. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 770–778 (2016)
  10. Hore, A., Ziou, D.: Image quality metrics: Psnr vs. ssim. In: 2010 20th International Conference on Pattern Recognition, pp. 2366–2369. IEEE (2010)
  11. Hu, J., Shen, L., Sun, G.: Squeeze-and-excitation networks. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 7132–7141 (2018)
  12. Huang, G., Liu, Z., Van Der Maaten, L., Weinberger, K.Q.: Densely connected convolutional networks. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 4700–4708 (2017)
  13. Hui, T.W., Tang, X., Loy, C.C.: Liteflownet: A lightweight convolutional neural network for optical flow estimation. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 8981–8989 (2018)
  14. Jin, M., Meishvili, G., Favaro, P.: Learning to extract a video sequence from a single motion-blurred image. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 6334–6342 (2018)
  15. Kim, T.H., Lee, K.M.: Segmentation-free dynamic scene deblurring. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 2766–2773 (2014)
  16. Kim, T.H., Lee, K.M., Scholkopf, B., Hirsch, M.: Online video deblurring via dynamic temporal blending network. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 4038–4047 (2017)
  17. Kim, T.H., Nah, S., Lee, K.M.: Dynamic video deblurring using a locally adaptive blur model. IEEE Transactions on Pattern Analysis and Machine Intelligence **40**(10), 2374–2387 (2018). DOI 10.1109/TPAMI.2017.2761348
  18. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)
  19. Kupyn, O., Budzan, V., Mykhailych, M., Mishkin, D., Matas, J.: Deblurgan: Blind motion deblurring using conditional adversarial networks. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 8183–8192 (2018)
  20. Lee, H.S., Kwon, J., Lee, K.M.: Simultaneous localization, mapping and deblurring. In: 2011 International Conference on Computer Vision, pp. 1203–1210. IEEE (2011)
  21. Lee, H.S., Lee, K.M.: Dense 3d reconstruction from severely blurred images using a single moving camera. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 273–280 (2013)
  22. Lei, T., Zhang, Y., Wang, S.I., Dai, H., Artzi, Y.: Simple recurrent units for highly parallelizable recurrence. arXiv preprint arXiv:1709.02755 (2017)
  23. Lin, M., Chen, Q., Yan, S.: Network in network. arXiv preprint arXiv:1312.4400 (2013)
  24. Michaeli, T., Irani, M.: Blind deblurring using internal patch recurrence. In: European Conference on Computer Vision, pp. 783–798. Springer (2014)
  25. Nah, S., Baik, S., Hong, S., Moon, G., Son, S., Timofte, R., Lee, K.M.: Ntire 2019 challenge on video deblurring and super-resolution: Dataset and study. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, pp. 0–0 (2019)
  26. Nah, S., Kim, T.H., Lee, K.M.: Deep multi-scale convolutional neural network for dynamic scene deblurring. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 3883–3891 (2017)
  27. Nah, S., Son, S., Lee, K.M.: Recurrent neural networks with intra-frame iterations for video deblurring. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 8102–8111 (2019)
  28. Nair, V., Hinton, G.E.: Rectified linear units improve restricted boltzmann machines. In: Proceedings of the 27th international conference on machine learning (ICML-10), pp. 807–814 (2010)
  29. Niklaus, S., Mai, L., Liu, F.: Video frame interpolation via adaptive separable convolution. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 261–270 (2017)
  30. Nimisha, T.M., Kumar Singh, A., Rajagopalan, A.N.: Blur-invariant deep learning for blind-deblurring. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 4752–4760 (2017)
  31. Pan, J., Bai, H., Tang, J.: Cascaded deep video deblurring using temporal sharpness prior. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 3043–3051 (2020)
  32. Ren, W., Pan, J., Cao, X., Yang, M.H.: Video deblurring via semantic segmentation and pixel-wise non-linear kernel. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 1077–1085 (2017)
  33. Rim, J., Lee, H., Won, J., Cho, S.: Real-world blur dataset for learning and benchmarking deblurring algorithms. In: European Conference on Computer Vision, pp. 184–201. Springer (2020)
  34. Schuler, C.J., Christopher Burger, H., Harmeling, S., Scholkopf, B.: A machine learning approach for non-blind image deconvolution. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1067–1074 (2013)
  35. Shen, W., Bao, W., Zhai, G., Chen, L., Min, X., Gao, Z.: Blurry video frame interpolation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 5114–5123 (2020)
  36. Son, H., Lee, J., Lee, J., Cho, S., Lee, S.: Recurrent video deblurring with blur-invariant motion estimation and

- pixel volumes. *ACM Transactions on Graphics (TOG)* **40**(5), 1–18 (2021)
37. Su, S., Delbracio, M., Wang, J., Sapiro, G., Heidrich, W., Wang, O.: Deep video deblurring for hand-held cameras. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1279–1288 (2017)
  38. Sun, D., Yang, X., Liu, M.Y., Kautz, J.: Pwc-net: Cnns for optical flow using pyramid, warping, and cost volume. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 8934–8943 (2018)
  39. Sun, J., Cao, W., Xu, Z., Ponce, J.: Learning a convolutional neural network for non-uniform motion blur removal. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 769–777 (2015)
  40. Sun, L., Cho, S., Wang, J., Hays, J.: Good image priors for non-blind deconvolution. In: European Conference on Computer Vision, pp. 231–246. Springer (2014)
  41. Tao, X., Gao, H., Shen, X., Wang, J., Jia, J.: Scale-recurrent network for deep image deblurring. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 8174–8182 (2018)
  42. Wang, X., Chan, K.C., Yu, K., Dong, C., Change Loy, C.: Edvr: Video restoration with enhanced deformable convolutional networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, pp. 0–0 (2019)
  43. Wang, Y., Lu, Y., Gao, Y., Wang, L., Zhong, Z., Zheng, Y., Yamashita, A.: Efficient video deblurring guided by motion magnitude. arXiv preprint arXiv:2207.13374 (2022)
  44. Wieschollek, P., Hirsch, M., Scholkopf, B., Lensch, H.: Learning blind motion deblurring. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 231–240 (2017)
  45. Wu, Y., Ling, H., Yu, J., Li, F., Mei, X., Cheng, E.: Blurred target tracking by blur-driven tracker. In: 2011 International Conference on Computer Vision, pp. 1100–1107. IEEE (2011)
  46. Wulff, J., Black, M.J.: Modeling blurred video with layers. In: European Conference on Computer Vision, pp. 236–252. Springer (2014)
  47. Xu, L., Jia, J.: Two-phase kernel estimation for robust motion deblurring. In: European conference on computer vision, pp. 157–170. Springer (2010)
  48. Xu, L., Ren, J.S., Liu, C., Jia, J.: Deep convolutional neural network for image deconvolution. *Advances in neural information processing systems* **27**, 1790–1798 (2014)
  49. Yu, Z., Liu, G.: Sliced recurrent neural networks. arXiv preprint arXiv:1807.02291 (2018)
  50. Zhang, X., Chen, Q., Ng, R., Koltun, V.: Zoom to learn, learn to zoom. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 3762–3770 (2019)
  51. Zhang, Y., Tian, Y., Kong, Y., Zhong, B., Fu, Y.: Residual dense network for image super-resolution. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 2472–2481 (2018)
  52. Zhang, Y., Tian, Y., Kong, Y., Zhong, B., Fu, Y.: Residual dense network for image restoration. *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2020)
  53. Zhong, Z., Gao, Y., Zheng, Y., Zheng, B.: Efficient spatio-temporal recurrent neural network for video deblurring. In: European Conference on Computer Vision, pp. 191–207. Springer (2020)
  54. Zhong, Z., Sun, X., Wu, Z., Zheng, Y., Lin, S., Sato, I.: Animation from blur: Multi-modal blur decomposition with motion guidance. arXiv preprint arXiv:2207.10123 (2022)
  55. Zhong, Z., Zheng, Y., Sato, I.: Towards rolling shutter correction and deblurring in dynamic scenes. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 9219–9228 (2021)
  56. Zhou, S., Zhang, J., Pan, J., Xie, H., Zuo, W., Ren, J.: Spatio-temporal filter adaptive network for video deblurring. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 2482–2491 (2019)
  57. Zhu, X., Hu, H., Lin, S., Dai, J.: Deformable convnets v2: More deformable, better results. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 9308–9316 (2019)
  58. Zoran, D., Weiss, Y.: From learning models of natural image patches to whole image restoration. In: 2011 International Conference on Computer Vision, pp. 479–486. IEEE (2011)