A Residual Learning Approach to Deblur and Generate High Frame Rate Video with an Event Camera

Haoyu Chen, Minggui Teng, Boxin Shi, Senior Member, IEEE, Yizhou Wang, and Tiejun Huang, Senior Member, IEEE

Abstract—Event cameras are bio-inspired cameras that can measure the intensity change asynchronously with high temporal resolution. One of the advantages of event cameras is that they suffer less from motion blur than traditional frame cameras when recording daily scenes with fast-moving objects. In this paper, we formulate the deblurring task on traditional cameras directed by events to be a residual learning one, and propose corresponding network architectures for effective learning of deblurring and high frame rate video generation tasks. We first train a modified U-Net network to restore a sharp image from a blurry image using the corresponding events. Then we train another similar network by replacing the downsampling blocks with blocks of the convolutional long short-term memory (ConvLSTM) to recurrently generate high frame rate video using the restored sharp image and part of the events. Benefitting from the blur-free events and the proposed learning strategy, the experimental results show that the proposed method outperforms state-of-the-art methods for generating sharp images and high frame rate videos.

Index Terms—Deblur, HFR Video Generation, Event Camera, Residual Learning.

I. INTRODUCTION

Event cameras, such as the Dynamic Vision System (DVS) [6], the Dynamic and Active-pixel Vision Sensor (DAVIS) [7], ATIS [8], and CelePixel [9] are bio-inspired sensors that asynchronously detect the change in log intensity of each pixel independently, unlike traditional frame cameras that sample the intensity of each pixel during the exposure time to create an image. When the intensity change reaches a threshold, an event is triggered that includes the position (X- and Y- coordinates), timestamp, and polarity information. Thus, the outputs of event cameras are not frames but a sequence of events.

Event cameras have several advantages over traditional cameras, such as higher temporal resolution (around 1 μs) and higher dynamic range (140 dB). As a result, event cameras do not suffer from many problems of traditional cameras, especially motion blur, which remains a challenging problem in computer vision and image processing.

Motion blur is the result of the relative motion between the camera and the scene during the image integration time, such as camera shake or object movements. Traditional methods to restore a sharp image from a blurry one apply various constraints to model the blur characteristics (e.g., non-blind uniform, blind uniform, and non-uniform deblurring) and utilize different natural image priors to regularize the solution space [10]–[16]. Most of these methods [11]–[13], [15] involve computationally intensive approaches, heuristic methods, and parameter tuning for determining the blur kernel parameters with a natural prior and lack a stable performance for many real scenes. Recently, learning-based methods have been proposed for deblurring, exhibiting a bright prospect in this field. Early methods employed traditional models, and most substituted neural networks for the blur kernel estimation operators [17]–[19]. Recent methods have designed end-to-end network structures for image deblurring [1], [2], [20], performing blind deblurring without estimating the blur kernels and directly generating deblurred images.

Event cameras have been used in various computer vision tasks, such as high dynamic range (HDR) imaging [21], [22], face detection [23], object classification [24], and optical flow estimation [25] by converting events to images [21], [25] or using events directly [22]–[24] in the inference problem. Event cameras are well suited for motion deblurring tasks due to the image acquisition method that depends on contrast change, for instance, motion-induced change. By combining complementary information from images and events [5], [26]–[28], motion deblurring can be achieved with high perceptual quality and low computational cost for model convergence.

In this paper, we propose a learning-based method that incorporates events to restore a sharp image from a blurry image and generate a high frame rate (HFR) video. Unlike existing works that generate sharp images directly using events, we formulate the deblurring task as a residual learning task and propose a modified U-Net [29] architecture with DenseNet [30] blocks in which event stacks are transformed into a log-difference image. In our work, the log-difference image is the difference between a blurry image and a sharp
image in the log space, which is approximately linear with event integration, providing useful cues to bridge the blurry image and sharp image. The sharp image is restored by adding the log-difference image to the blurry image. After deblurring, we combine the restored sharp image with the events to generate an HFR video using a similar residual learning network architecture, in which the DenseNet blocks are replaced by the blocks of the convolutional long short-term memory (Conv-LSTM). The contributions of this paper include proposing:

- a linear forward model linking blurry and sharp images with log-difference images;
- a residual architecture for effective learning the log-difference images; and
- a unified framework for event-based deblurring and HFR video generation.

We quantitatively and qualitatively evaluate the proposed residual network on synthetic and real datasets and demonstrate that the proposed method outperforms state-of-the-art methods.

II. RELATED WORK

A. Learning-based image and video deblurring

Deep learning has proven effective in many computer vision applications such as object detection [31]–[33], segmentation [34], and image and video deblurring [1]–[3], [17], [35]–[38]. A neural network derives prior information from the training data and restores sharp images from corresponding features extracted from blurry images.

State-of-the-art learning-based deblurring methods use a single frame or multiple frames. For single image deblurring, end-to-end methods generating deblurring images directly were proposed. Nah et al. [1] proposed a multi-scale convolutional neural network (CNN) that directly restores latent images without using a restricted blur kernel model or estimating explicit blur kernels. Further, Tao et al. [2] proposed a multi-scale encoder-decoder network. The blurry image was restored at different resolutions from coarse to fine in the scale-recurrent network. Kupyn et al. [3] used the popular generative adversarial network (GAN) for deblurring. Liang et al. [39] leverages informative raw images to address the image deblurring problem. Li et al. [40] proposed a deep DEBLUR-IQA to guide the deblurring network’s optimization. However, although these methods considered real-world blur motion, it was challenging to obtain sharp images. Some important motion information cannot be extracted from a single image. Therefore, video deblurring that considers temporal information in consecutive frames represents a potential approach.

Jin et al. [35] proposed a multi-frame deblurring network to extract a video from blurry images by restoring the middle sharp image and calculating temporal ambiguities. Zhang et al. [36] proposed 3D convolution in the spatial and temporal domains to extract motion features from video data and used a GAN to sharpen the images and improve their appearance. Nah et al. [37] incorporated the hidden states extracted from past frames to the current frame to utilize motion information from consecutive video frames. Zhou et al. [38] used a spatio-temporal filter adaptive network for feature alignment and deblurring. Their model recurrently combines information from the previous frame and the current frame to generate deblurred videos. In these methods, the temporal information in the blurry videos provides prior information for the deblurring task, significantly improving the image quality. However, videos captured by traditional cameras have a relatively low time resolution. Therefore, we use event cameras to exploit temporal information.
B. High frame rate video generation methods

Early HFR video generation methods focused on video interpolation with optical flow. After estimating the optical flow from two input frames, intermediate frames can be generated at arbitrary times between them. However, motion boundaries and severe occlusions are significant challenges. The success of deep learning has inspired numerous deep learning models for HFR video generation. Benefiting from the neural network, they did not obtain the optical flow explicitly. Niklaus et al. [41] formulated the frame interpolation as a local convolution over the two input frames and used a CNN to derive a spatially-adaptive convolution kernel for each pixel. The interpolation frame was restored by the convolution operation. However, the method is computationally and memory intensive since it predicts a kernel for each pixel. Liu et al. [42] proposed a CNN model for frame interpolation with an explicit sub-network for motion estimation. Excellent interpolation results were obtained due to the end-to-end training methods.

However, the CNN-based single-frame interpolation methods [41], [42] are not well-suited for multi-frame interpolation. Although they can recurrently interpolate frames, the error accumulation results in low quality of the restored frames. Jiang et al. [43] proposed the Super SloMo network to interpolate arbitrary frames between consecutive frames by estimating the optical flow to the intermediate frame. However, linear motion between the consecutive frames was assumed, which may not be suitable for real-world applications. In contrast, our method uses the event sequence to produce HFR videos because the motion information is captured in the event sequence since its time resolution can achieve micro-second level.

C. Event-based methods

Several works combined events captured by event cameras (events might also refer to a kind of feature representation extracted from video [44], [45], which is beyond the scope of this paper.) with intensity frames and reconstructed high-quality images and HFR videos. Reinbacher et al. [22] proposed a variational model for reconstructing intensity images with intensity frames and reconstructed high-quality images. Scheerlinck et al. [27] proposed a continuous-time formulation of event-based intensity estimation using complementary filtering to combine image frames with events and obtain continuous-time image intensities. Mohammad et al. [21] used an event-based conditional GAN (cGAN) [46] to create images/videos from an adjustable portion of the event data stream based on the spatio-temporal intensity changes. These methods utilized specific information in the event sequence to solve some computer vision problems, providing excellent results that demonstrated the advantage of event cameras.

Events were also used for deblurring and HFR video generation because of the high temporal resolution and reliable motion information encoded in the captured events. Pan et al. [5] proposed an optimization model (event-based double integral (EDI)) for estimating a single scalar variable and restoring a sharp image. An HFR video was generated from the restored image using the optimized scalar, which represented the threshold for triggering the event. Rebecq et al. [47] proposed a U-Net-like network with Conv-LSTM blocks and generated a video directly from encoded events using a spatio-temporal voxel grid. As shown in Figure 1, these two methods result in the rapid accumulation of errors. Pan et al. [5] did not fully consider noisy events and formulate the physical model with a constant threshold. And Rebecq et al. [47] only utilized the event sequences, resulting in a lack of texture in the background. Jiang et al. [28] used a convolutional recurrent neural network that integrates spatial and temporal knowledge of both global and local scales to recover image details. Lin et al. [48] proposed a deep CNN with a dynamic filtering layer to deblur and generate HFR videos. Wang et al. [49] proposed an event-enhanced sparse learning network named eSL-Net to address deblurring, denoising, and SR simultaneously. However, their method did not make full use of the connection between the events and blurry images. We utilize the strong representation power of the residual model; thus, our model is not affected strongly by noisy events, and avoids rapid accumulation of errors.

III. PROPOSED METHOD

In this section, we briefly introduce the image acquisition method of event cameras (Section III-A). We then present the proposed learning-based methods for the deblurring task (Section III-B) and the design methodology of the network (Section III-C). We propose a method to generate a sequence of frames using the restored image (output of the deblurring task) and the related events (Section III-D). We compare our method with existing methods that restore and generate images using events (Section III-E).

A. Event camera image acquisition

Unlike traditional cameras, an event camera records the intensity change of each pixel individually. If the intensity change reaches a threshold $C$, the camera triggers an event, $e = \{u, t, p\}$, where $u = (x, y)^T$ is the coordinate of the pixel, $t$ is the timestamp of event, and $p$ is the polarity of the event. The change in log intensity is measured for a pixel $u$ with intensity $I(t_0)$. If at time $t$, the log intensity $L = \log I$ changes beyond the threshold $C$ in a period $\Delta t$, i.e.,

$$|L(u, t) - L(u, t - \Delta t)| \geq C,$$

(1)

the camera outputs an event $e = (u, t, p); p = 1$ (or $-1$) if the difference $L(u, t) - L(u, t - \Delta t)$ is greater (or less) than 0.

B. Event-based residual image formation model

1) Image deblurring model: We first link the event sequence from an event camera and the images to restore the sharp image. Given the intensity of sharp images, as described in [1], blur accumulation process can be modeled as:

$$B(u) = \frac{1}{T - t_0} \int_{t_0}^{T} I(u, t) dt \approx \frac{1}{T - t_0} \sum_{t=t_0}^{T} I(u, t).$$

(2)

Equation 2 applies to each pixel $u$ independently, and subscripts $u$ denoting pixel location are omitted henceforth.
If \( \epsilon(t_1, t) = \{e_i\}_{i=1}^{N_e} \) is the incoming event stream at a pixel from \( t_1 \) to \( t \), the latent image at \( t \) can be expressed as:

\[
I(t) = I(t_1) \cdot \exp \left( \sum_{i=1}^{N_e} C_i \right),
\]

where \( C_i \) is the threshold of the intensity change to trigger the \( i \)-th event \( e_i \). The threshold is not constant but has different values for positive and negative intensity changes, following a normal distribution over time [50]. It is non-trivial to establish an analytical expression for the change caused by \( e_i \) at time \( t \); thus, we denote it as \( f(e_i, I(t)) \). We estimate the change by triggering an event using the local intensity and gradient information.

By combining Equation 2 and Equation 3, we obtain:

\[
\begin{align*}
B & = \frac{1}{T - t_0} \sum_{t = t_0}^{T} I(t_k) \cdot \exp \left( \sum_{i=1}^{N_e} C_i \right) \\
& = \frac{1}{T - t_0} \sum_{t = t_0}^{T} I(t_k) \cdot \prod_{i} \exp(C_i) \\
& = \frac{1}{T - t_0} \sum_{t = t_0}^{T} I(t_k) \cdot \prod_{i} f(e_i, I(t)),
\end{align*}
\]

where \( t_k \in [t_0, T] \), and \( \{e_i\}_{i=1}^{N_e} \) is triggered between \( t_k \) and \( t \). We apply the logarithm on both sides of Equation 4, where \( L = \log I \), and rearrange the equation to obtain:

\[
L(t_k) = \log B - \log \left( \frac{1}{T - t_0} \sum_{t = t_0}^{T} \prod_{i} f(e_i, I(t)) \right). \tag{5}
\]

The intensity value of the sharp image \( I(t) \) is unavailable in the deblurring task. However, in a blurry image, gradient information can be observed along with the edge of the low-level blurry part. Even for some serious blurry areas, the gradient can be partially extracted. The sharp and blurry regions are similar in the less blurry background (low-frequency part). Thus, we approximately use \( B \) as a proxy of \( I(t) \) for \( f(e_i, I(t)) \) in Equation 5 and obtain:

\[
L(t_k) = \log B - \log \left( \frac{1}{T - t_0} \sum_{t = t_0}^{T} \prod_{i} f(e_i, B) \right). \tag{6}
\]

2) HFR video generation model: Using a residual term encoded with events, Equation 6 provides the mapping from a blurry image \( B \) to an arbitrary latent frame \( L(t_k) \). The restored sharp image and subsequent events are used to generate the HFR video due to the high temporal resolution of the events. We use the log representation of Equation 3 to obtain the residual relationship between two adjacent frames:

\[
L(t_{k+1}) = L(t_k) + \sum_{i=1}^{N_e} \log \left( f(e_i, L(t_k)) \right), \quad (k = 0, \cdots, N - 1).
\]

We can apply Equation 6 to generate \( L(t_0) \) and Equation 7 to generate the \( N - 1 \) following images \( \{L(t_1), \cdots, L(t_N)\} \), which are combined to create the HFR video.

Equations 6 and 7 provide the mapping from an input image to an output image using a residual term encoded with events. The residual term is approximately linearly related to the event integration, providing information to predict the difference between the input and output images instead of generating images directly. We refer to the residual term as the log-difference image; the objective is to predict the image \( M_1 \) for deblurring and \( M_2 \) for HFR video generation. Since the threshold is not constant [50] and the function \( f(e_i, B) \) cannot be explicitly described, we propose two networks in the following subsections to learn the log-difference image guided by the input image.

C. Event-based residual deblurring network

1) Network architecture: The objective is to restore a sharp image using the events (Equation 6). As discussed above, we designed a learning-based method to derive the function using training data. We propose a deblurring network with a global skip connection, as shown in “Deblurring Stage” of Figure 2. Given the blurry image and events, the log-difference image is predicted as follows:

\[
M_1(\epsilon, B, \theta) = \log \left( \frac{1}{T - t_0} \sum_{t = t_0}^{T} \prod_{i} f(e_i, B) \right), \tag{8}
\]

where \( M_1(\epsilon, B, \theta) \) is the output log-difference image of the deblurring network trained with the event sequence \( \epsilon \), the blurry image \( B \), and \( \theta \), which consists of all network parameters to be learned. We obtain a sharp image in the log space by adding \( M_1 \) to the blurry image.

According to the results in Pan et al. [5], gradient features can guide the deblurring task effectively. The proposed network has to learn the relationship between the event sequence and the intensity change depending on the local information and the features extracted from the blurry image. The deblurring network has a U-Net-like structure that considers the physical model of the event-based deblurring method. The deblurring network consists of three parts: the Dense Encoder module, which extracts features from the blurry images and stacked events bins separately, the Denoiser module, which removes noisy events located at the blurry edges, and the Decoder module, which fuses the encoded blurry images with the encoded stacked events.

**Dense Encoder.** The Dense Encoder module consists of four downsampling layers, which is a dense convolution block followed by a \( 2 \times 2 \) downsampling convolution. The dense convolution block is designed based on DenseNet [51] to extract local features, which may be the gradient of the blurry image and the gradient of the current estimate, possibly thresholded in the low-frequency regions [52]. Recent work has shown that DenseNet can be trained to extract high-level features from an image better than the popular ResNet [53]. We train our encoder with stacked event frames to extract features from events in a neighborhood. In addition, we feed the blurry images and stacked events into the network at two entrances for full use of the distinct information in the events (e.g., motion clues) and the blurry images (e.g., the intensity value) instead of concatenating them. A detailed discussion can be found in Section IV-E1.
Fig. 2: The pipeline of the proposed method to restore an HFR video from a blurry image. In the Deblurring Stage, we use the event-based deblurring residual network with all events \( V(0 : N) \) and the blurry image \( B \) to obtain the deblurred image \( \mathbf{L}(t_0) \). In the HFR Video Generation Stage, we use the event-based HFR video generation residual net with part of the events and the output frame \( \mathbf{L}(t_0) \) to predict the next frame \( \mathbf{L}(t_1) \). This process is used repeatedly to obtain the HFR video. For better visualization, we assume that only 4 sharp frames \( (N = 3) \) are recovered from the blurry image. The detailed layer and parameter configurations can be found in the supplementary material.

**Denoiser.** Spurious events, caused by the circuit or environment change, broadly exist in event sequence, and significantly degrade the image quality (we assume events are mainly triggered by object motion or camera shake, and ignore the events caused by illumination change like Pan et al. [5], who ignore glare and no-light conditions). Thus, we need to filter noise from the event sequence, which is located along the blurry edges and not removed by the activation functions. If the noisy events along the edges are not removed, the noise is amplified in the decoder part; an example is shown in Figure 3. Xie et al. [54] observed that adversarial perturbations of images led to noise in the image features. Therefore, we propose feature denoising to improve the image quality. We also formulate the denoising procedure as a residual learning task. Guided by the extracted features from the intensity images, the proposed network, consisting of 5 residual convolutional layers, can suppress majority of the noise, resulting in sharp yet photometrically consistent edges.

**Decoder.** The Decoder module consists of three upsam-
plugging layers, i.e., dense deconvolution blocks, a feature fusion block to concatenate the upsampling features with the skip-connected features, and a prediction layer. The output of the Decoder module is the log-difference image \( M_1 \), which is obtained by combining the sequence of events and the blurry images. Especially with a larger convolutional kernel in the last prediction layer. The decoder module considers the correlation among neighboring pixels and assigns different weights to the events in the sequence during log-difference image generation. The weights correspond to the variant thresholds in different regions.

2) Data input: The output of event cameras is a sequence consisting of event tuples \( (u, t, p) \), as described in Section III-A. A representation of the events is required to feed them into the network. We adopt the representation in [25] and establish a 3D event volume by merging and stacking the events in a small time interval. Given an event sequence \( \{e_1, e_2, \cdots, e_n\} \) from time \( t \) to \( T \) at a pixel location \( (x, y) \), we discretize the time dimension into \( N \) bins. The 3D event volume \( V \) corresponding to pixel \( (x, y) \) can be obtained as follows:

\[
t_i = t + i \times \frac{T-t}{N}, i \in [0, N],
\]

\[
V_{x,y}(i) = \sum_{t_k \in \{t, t+1\}} p_k, (t_k, p_k \in e_k).
\]

The event volume \( V(0 : N) \) and the blurry image \( B \) are separately fed into the encoder. Moreover, as described in [25], stacking events into multiple bins is preferable to using one bin. More details of the input data will be described in Section IV-C.

3) Loss function: We define the loss function as follows:

\[
L(I_B, I_S) = L_1(I_B, I_S) + \lambda \cdot L_{PL}(I_B, I_S),
\]

where \( I_S \) denotes the generated images and \( I_B \) denotes the blurry images. \( L_1(I_B, I_S) \) is the mean absolute error (MAE) loss between \( I_B \) and \( I_S \). The perceptual loss \( L_{PL} \) [55] is an improvement of \( L_2 \)-loss based on VGG19 trained on ImageNet. It does not calculate the difference between the generated images and the target images but uses the average of the differences in the feature maps. The perceptual loss measures high-level perceptual and semantic differences between blurry images and sharp images and provides more robust constraints than the \( L_2 \)-loss. It is also more sensitive to the blurry than the \( L_2 \)-loss. The perceptual loss can be defined as follows:

\[
L_{PL}(I_B, I_S) = \frac{1}{C_j W_j H_j} \| \phi_j(I_B) - \phi_j(I_S) \|_2^2,
\]

where \( \phi_j(I_B) / \phi_j(I_S) \) is the feature map of \( I_B / I_S \) of shape \( C_j \times W_j \times H_j \), obtained from the \( j \)-th VGG19 network convolution layer. I.e., the perceptual loss is Euclidean distance between feature representations of \( I_B \) and \( I_S \), encouraging them to have similar feature representations. The layers we used to compute perceptual loss are 4-th, and 5-th convolution layers of the VGG19 network in our experiments.

D. Event-based residual HFR video generation network

1) Network architecture: Since Equations 6 and 7 have a similar form, we can design a residual learning network for HFR video generation. Given the events and the previous frame, the log-difference image is predicted as follows:

\[
M_2(e(t_k, t_{k+1}), L(t_k), \theta) = \sum_{i=1}^{N_c} \log f(e_i, L(t_k)).
\]

A straightforward design of this network is using events and one frame as input and the output is the next frame. However, experiments indicate that it is difficult to train this network, and low performance is obtained because of the rapid error accumulation (the generated image quality decreases significantly after the third frame). Therefore, we use a residual learning network for the video generation, as proposed in Section III-B.

LSTM Encoder. The network structure for this task is similar to the deblurring task, as depicted in the “HFR Video Generation Stage” in Figure 2. The major difference is that we replace the Dense Encoder with the LSTM Encoder, which has Conv-LSTM blocks [56] in downsampling layers, to learn the temporal information in the event sequence. The reason is that temporal consistency is the key in video generation, as described in [47]. Moreover, without the Conv-LSTM blocks in the frame generation process, the network would suffer from vanishing gradients during backpropagation because of the small differences between the adjacent frames. The proposed structure solves this problem and generates long sequences stably.

The pipeline of the proposed method to restore the HFR video from a blurry image is described in Algorithm 1 (corresponding to the “Deblurring Stage + HFR Video Generation Stage” in Figure 2). The event-based deblurring residual network uses a blurry image \( B \) captured from \( t_0 \) to \( T \), and all corresponding events \( V(0 : N) \) to restore a sharp image, which is used as the first frame at the HFR video generation stage. Next, the event-based HFR video generation residual network recurrently uses a part of the events and a sharp frame to generate the next frame, i.e., to obtain the frame \( L(t_{k+1}) \). By discretizing the event into \( N \) bins, we can obtain \( N \) + 1 latent frames from a blurry image.

**Algorithm 1** Deblurring and HFR video generation with events

Require: the blurry image \( B \), events \( e(t_0, T) \)
1: Divide events into \( N \) bins \( V(0 : N) \)
2: Deblurring with all events using Equation 6 and Equation 8
3: \( \ell(t_0) = \log B - M_1(V(0 : N), B, \theta_1) \)
4: for \( k = 0 \) to \( N - 1 \) do
5: Get \( L(t_k) \) with part of events \( e(t_k, t_{k+1}) \) from Equation 7 and Equation 13
6: \( L(t_{k+1}) = L(t_k) + M_2(V(k), L(t_k), \theta_2) \)
7: end for
8: return HFR video \( L = \{L(t_0), L(t_1), \cdots, L(t_N)\} \)
2) Data input: The inputs of the HFR video generation network are a sharp image and stacked events during the theoretical exposure time of the image to be generated. The first sharp image is the restored image obtained from the event-based deblurring residual network. Then we repeatedly use the generated frame as the input to the next step. As described in Section III-C, we also divide the input events into several bins. The time resolution of event cameras represents the time interval of consecutive events in the stream, and it is different from the traditional videos. In theory, it can achieve the micro-second level [7], which is sufficient to improve the frame rate. For our experiment, we choose to divide the events in the synthetic dataset into 6 bins and obtain HFR videos with 240 FPS for fair comparisons with the other state-of-the-art methods. The real data are divided into 30 bins, resulting in a 30× higher frame rate (theoretical upper bound) than the original blurry video. The loss function in Equation 11 is used for this task.

E. Relationship with existing models

The EDI model described in [5] can be regarded as a special case of our model if \( C_i \) is a constant. It formulated the deblurring task with events to be a non-convex optimization problem with the Fibonacci search method and a constant threshold is used to restore the sharp image. In contrast, in our model, the threshold in the physical model of the event camera is not a constant due to environmental influences. A constant global threshold produces errors and artifacts in the restored image, as shown in Figure 4(b).

The learning-based model proposed in [47] also uses events to generate an HFR video. Since only events were used as input, the background and texture information is missing in the generated video. The proposed residual network for HFR video generation combines the restored clear background of the intensity image with the event sequence, generating a sharp video with rich background details, as shown in Figure 4(c) and (d). More comparisons will be provided in Section IV-D.

IV. Experiments and Evaluation

In this section, we compare our method with several state-of-the-art methods for image deblurring and HFR video generation. We use synthetic and real datasets and conduct qualitative and quantitative evaluations.

A. Data preparation

1) Synthetic dataset: For the quantitative comparison of the experiment results of the proposed and other deblurring methods, we generate a synthetic dataset based on the GoPro dataset [1]. This dataset consists of 240 FPS videos taken with GoPro cameras. We average 7 successive frames to produce blurry images (different from Nah et al. [1], which average different numbers of successive latent frames (7∼13) to produce blurred images) and generate events between two consecutive images using the event simulator provided by Rebecq et al. [50] with non-uniform thresholds.

Fig. 4: An example of the deblurring result. (a) A blurry image. (b)∼(d) The restored sharp images by Pan et al. [5], Rebecq et al. [47], and ours. Artifacts are observed, such as noise around the restored letters (b) and oversmoothing of the folds (c).

2) Real dataset: The real dataset was obtained by the authors and from [5]. The frames and events are captured by a DAVIS240 event camera. Different scenes are used, and different motion patterns (e.g., camera shake, objects motion) are included to obtain intensity images with motion blur.

B. Evaluation metrics

For the evaluation metrics, we use the Peak Signal-to-Noise Ratio (PSNR) and the Structural Similarity Index (SSIM) between the generated and ground-truth images. These metrics are typically used for quantitative comparisons. However, the PSNR and SSIM have some limitations. A well-known example is that blur causes large perceptual change but not much PSNR change. In our network training procedure, the perceptual loss is a significant indicator of the level of blurriness. Therefore, we adopt the Learned Perceptual Image Patch Similarity (LPIPS) [57], which is based on a network trained on a large distortion dataset, such as deblurring distortion, and CNN-based distortion. The LPIPS assesses the perceptual similarity between the ground truth and the deblurred image; a smaller LPIPS indicates less image distortion or less blurriness. We can evaluate the experimental results more comprehensively by combining the three metrics.

C. Training details

The network training is conducted using an NVIDIA GeForce GTX 1080Ti GPU and implemented on the PyTorch platform. The optimizer is ADAM, and the learning rate is initialized at 0.002 and decreased by a factor of 10 every 30 epochs. In our experiment, we find that 70 epochs (45220 iterations) are sufficient for model convergence. During each epoch, we use a batch of 5 blurry images and related events.

Since the frame rate of a DAVIS event camera is less than 30 FPS, it is difficult to generate ground-truth deblurred images using this; thus, we only use the synthetic dataset.
to train the network. The GoPro dataset is divided into two parts with a 2:1 ratio. 90% of the training dataset are randomly chosen as training samples and the rest 10% are used for validation. During the training process of the event-based residual deblurring network, an image with a size of 256×256, whereas the image in the HFR video generation network is cropped to 128×128. As there are enough events in most areas in the dataset (at least 10 thousand events in the corresponding time duration in the experiment), we randomly crop the events from the entire image. Moreover, we convert the color images in the GoPro dataset to grayscale images because the real dataset only contains grayscale images. For generating color images in the test procedure, we use synthetic events in each channel of the images, which is consistent with the principle of a color DVS [60]. Then we restore the three channels separately and concatenate them for the final output. Our network also keeps the consistency of three channels; thus, we do not need to align them.

Moreover, we divide the input events into six bins rather than just stacking them into one bin as described in [4], [21], [25]. The use of several bins results in more accurate temporal information. However, more event bins do not necessarily produce better results because it is more difficult to filter the noise. In our experiment, six bins resulted in higher image quality than one bin, i.e., PSNR of 34.25 dB vs. 29.93 dB and SSIM of 0.9534 vs. 0.9043, respectively. The number of event

Fig. 5: Examples of the deblurring results on the synthetic dataset derived from the GoPro dataset [1]. (a) The blurry input image. (b) Ground truth. (c)∼(l) Deblurring results of Nah et al. [1], Kupyn et al. [3], Tao et al. [2], Wang et al. [49], Jin et al. [35], Zhong et al. [58], Pan et al. [5], Mehri et al. [59], Jiang et al. [28], and our method. More results are provided in the supplementary material.
D. Experimental results

1) Deblurring: We compare the proposed method with five learning-based image deblurring methods [1]–[3], [59], [61], a video deblurring method [58], and three event-based deblurring methods [5], [28], [49] on the synthetic dataset. We also make the qualitative comparison with Rebecq et al. [47], as a reference for visual quality (note Rebecq et al.'s method does not output a frame). More results are provided in the supplementary material.

All learning-based deblurring methods are trained on GoPro dataset [1] and generate middle frames as deblurred ones. However, for the HFR video generation comparison, our deblurring network generates the first latent frame.
TABLE I: Quantitative results of deblurring on synthetic dataset. All methods are tested under the same blurry condition. ↑ (↓) indicates the higher (lower), the better throughout this paper. The best performances are highlighted in bold.

<table>
<thead>
<tr>
<th>Method</th>
<th>PSNR ↑</th>
<th>SSIM ↑</th>
<th>LPIPS ↓</th>
</tr>
</thead>
<tbody>
<tr>
<td>Nah et al. [1]</td>
<td>31.30</td>
<td>0.9113</td>
<td>0.0779</td>
</tr>
<tr>
<td>Tao et al. [2]</td>
<td>30.54</td>
<td>0.9240</td>
<td>0.0946</td>
</tr>
<tr>
<td>Kupyn et al. [3]</td>
<td>30.18</td>
<td>0.9064</td>
<td>0.0914</td>
</tr>
<tr>
<td>Purohit et al. [61]</td>
<td>30.58</td>
<td>0.9410</td>
<td>0.1376</td>
</tr>
<tr>
<td>Pan et al. [5]</td>
<td>30.68</td>
<td>0.9088</td>
<td>0.1389</td>
</tr>
<tr>
<td>Jiang et al. [28]</td>
<td>31.13</td>
<td>0.9147</td>
<td>0.1358</td>
</tr>
<tr>
<td>Wang et al. [49]</td>
<td>29.89</td>
<td>0.8912</td>
<td>0.1227</td>
</tr>
<tr>
<td>Zhong et al. [58]</td>
<td>31.07</td>
<td>0.9023</td>
<td>0.1563</td>
</tr>
<tr>
<td>Mehrj et al. [59]</td>
<td>33.62</td>
<td>0.9459</td>
<td>0.0786</td>
</tr>
<tr>
<td>Ours</td>
<td>34.25</td>
<td>0.9534</td>
<td>0.0726</td>
</tr>
</tbody>
</table>

The results for the real datasets are shown in Figure 1 and Figure 4 (captured by the DVS346 event camera), and Figure 6 (obtained from [5]). We can find clearly that our method restores a clearer image than the other learning-based methods. Compared with image-based methods [1]–[3], the results demonstrate that relying only on blurry images produced inferior deblurring results. For example, the results obtained by Nah et al. [1] show a poor generalization ability of the deblurring networks without additional information. The integration of additional information, such as events, improves the deblurring performance. A comparison with the EDI model [5], shows that the images obtained from our model suffer less from background noise than those obtained from the EDI model. Our restored images are smoother and clearer than those of Pan et al. [5], as discussed in Section IV-F.

2) **HFR video generation:** We compare our HFR video generation method with three recently proposed event-based generation methods [5], [28], [47] and a pure image-based method [43]. The results are shown in Figure 7 and Table II for synthetic dataset and in Figure 6 for the real dataset. Furthermore, we also compare the results with the method proposed in [43], i.e., video interpolation with pure visual frames, to demonstrate the advantage of using events.

Compared with image-based methods [35], [37], our method provides high quality of the restored images due to the low latency of the event signals. In comparison to event-based generation methods, the HFR video generated by our method has less noise and is more stable than the result obtained by [5]. Compared to the method in [47] that generates frames directly from event sequences, our restored images have richer details because the background information can be extracted from the blurry images. Jiang et al. [28] is also a learning-based method, while they formulate the problem as a denoising one, which is hard to learn with event sequence. Benefiting from our residual learning method, we can achieve higher PSNR and SSIM.

TABLE II: Quantitative results of HFR video generation on synthetic dataset.

<table>
<thead>
<tr>
<th>Method</th>
<th>PSNR ↑</th>
<th>SSIM ↑</th>
<th>LPIPS ↓</th>
</tr>
</thead>
<tbody>
<tr>
<td>Jin et al. [35]</td>
<td>28.01</td>
<td>0.8670</td>
<td>0.1523</td>
</tr>
<tr>
<td>Nah et al. [37]</td>
<td>29.97</td>
<td>0.8947</td>
<td>—</td>
</tr>
<tr>
<td>Pan et al. [5]</td>
<td>28.06</td>
<td>0.8617</td>
<td>0.1927</td>
</tr>
<tr>
<td>Jiang et al. [43]</td>
<td>28.69</td>
<td>0.8781</td>
<td>0.1240</td>
</tr>
<tr>
<td>Jiang et al. [28]</td>
<td>29.67</td>
<td>0.9270</td>
<td>—</td>
</tr>
<tr>
<td>Ours</td>
<td>31.89</td>
<td>0.9303</td>
<td>0.1256</td>
</tr>
</tbody>
</table>
We compare the state-of-the-art HFR video interpolation methods using our deblurring frames to demonstrate the advantage of using events. The results show that our method interpolates temporal frames with better quality. Due to the motion information in the events, we can estimate the object trajectory more accurately, facilitating the restoration of the latent frames. The video interpolation method [43] assumes uniform object motion velocity between two consecutive frames. Since the motion is non-uniform, the images restored by [43] have lower performance for the average result of the middle frames, i.e., PSNR of 31.63 dB vs. 27.99 dB and SSIM of 0.9284 vs. 0.8665. As shown in the error map in Figure 8, the image-based interpolation method [43] cannot precisely estimate the object motion trajectory. Our method shows better photometric consistency for generating the HFR video, whereas the method used in [43] causes the scene shaking, which is shown in the supplementary material.

We repeatedly use one deblurred image to generate 50 frames to determine the upper bound of the number of restored images. As shown in Figure 9, with a sufficient number of events, the proposed method can interpolate 20 frames with reasonably stable performance, given one image as input. When the number of interpolated frames exceeds 50, the performance drops because the photometric consistency cannot be maintained.

E. Model Analysis

1) Ablation studies: To prove the effectiveness of the network components, we conduct ablation experiments without a global connection (w/o GB), without a Denoiser (w/o DN), without the LSTM Encoder (w/o LSTM) in the HFR Video Generation Stage, and by replacing the two encoders with one encoder (w/o Two). w/o GB is used to show the benefit of using residual learning, w/o DN is evaluated to demonstrate the influence of event denoising, and w/o LSTM is used to show the effectiveness of Conv-LSTM; w/o Two is tested to determine the proper method to encode the events and images. The quantitative results of ablation studies are shown in Table III. For the HFR video generation, we focus on maintaining the image quality and photometric consistency of the frames; therefore, we also quantitatively evaluate the quality of the last interpolated frame (10 in this experiment).

**Effectiveness of Residual Learning.** Without the global residual connection, the network is regarded as an image-to-image translation network, and it is difficult to maintain the semantic information in the training procedure. Global residual connection formulates our network as a residual learning one, whose target is the residual term encoded with the events (log-difference image). Table III shows that the model w/o GB has the lowest performance based on all metrics, indicating the effectiveness of residual learning.

**Effectiveness of Two Encoders.** The proposed network extracts event features and image features separately in the encoding procedure to obtain a cleaner background. The full model exhibits a significant improvement in the PSNR and SSIM over the w/o Two model. Since the images and stacked events are separate inputs into the network, the intensity information in the blurry images and the edge or gradient information obtained from the stacked events are treated separately in the decoder part.

**Effectiveness of Denoiser.** The full model provides sharper images than the w/o DN model. Although the w/o DN model has relatively high values of PSNR and SSIM, the higher LPIPS than the full model indicates that the output images are distorted, especially near the edges. Noisy events, which occur near the blurry edges and are not removed by the activation functions, significantly degrade image quality. The residual block can remove the noisy events at the blurry edges, as shown in Figure 3.

**Effectiveness of LSTM Encoder.** Maintaining temporal information is crucial for HFR video generation. The proposed model uses the LSTM Encoder to encode the temporal information in the hidden states of the Conv-LSTM layer. Although the w/o LSTM model has PSNR and SSIM values comparable

<table>
<thead>
<tr>
<th>Average result of deblurring</th>
<th>PSNR ↑</th>
<th>SSIM ↑</th>
<th>LPIPS ↓</th>
</tr>
</thead>
<tbody>
<tr>
<td>w/o GB</td>
<td>28.93</td>
<td>0.9122</td>
<td>0.1500</td>
</tr>
<tr>
<td>w/o Two</td>
<td>32.71</td>
<td>0.9446</td>
<td>0.0963</td>
</tr>
<tr>
<td>w/o DN</td>
<td>33.90</td>
<td>0.9514</td>
<td>0.0908</td>
</tr>
<tr>
<td>Ours</td>
<td>34.25</td>
<td>0.9534</td>
<td>0.0726</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Average result of HFR video generation</th>
<th>PSNR ↑</th>
<th>SSIM ↑</th>
<th>LPIPS ↓</th>
</tr>
</thead>
<tbody>
<tr>
<td>All frames</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>w/o GB</td>
<td>28.53</td>
<td>0.8957</td>
<td>0.1859</td>
</tr>
<tr>
<td>w/o LSTM</td>
<td>31.34</td>
<td>0.9229</td>
<td>0.1284</td>
</tr>
<tr>
<td>Ours</td>
<td>31.89</td>
<td>0.9303</td>
<td>0.1256</td>
</tr>
<tr>
<td>Last frame</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>w/o GB</td>
<td>24.65</td>
<td>0.8079</td>
<td>0.2697</td>
</tr>
<tr>
<td>w/o LSTM</td>
<td>27.61</td>
<td>0.8788</td>
<td>0.1987</td>
</tr>
<tr>
<td>Ours</td>
<td>28.53</td>
<td>0.8957</td>
<td>0.1859</td>
</tr>
</tbody>
</table>
TABLE IV: Quantitative results of deblurring using the synthetic data containing blurry images and event stacks. All methods are tested under the same blurry condition. ↑ (↓) indicates the higher (lower), the better throughout this paper. The best performances are highlighted in bold.

<table>
<thead>
<tr>
<th>Method</th>
<th>Training (hours)</th>
<th>Execution (seconds)</th>
<th>Parameters (million)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Nah et al. [1]</td>
<td>0.20</td>
<td>11.10</td>
<td></td>
</tr>
<tr>
<td>Tao et al. [2]</td>
<td>72</td>
<td>0.72</td>
<td>3.76</td>
</tr>
<tr>
<td>Kupyn et al. [3]</td>
<td>120</td>
<td>0.35</td>
<td>2.65</td>
</tr>
<tr>
<td>Purohit et al. [61]</td>
<td>–</td>
<td>0.20</td>
<td>4.46</td>
</tr>
<tr>
<td>Ours</td>
<td>6</td>
<td>0.54 / 0.41</td>
<td>2.67</td>
</tr>
</tbody>
</table>

The best performances are highlighted in bold.

to our method, the visual quality of the images is relatively low based on the average PSNR and SSIM of the last frame. Meanwhile, the image distortion is also worse than for the proposed method.

2) Runtime Efficiency: The comparison of the runtimes of the different learning-based methods is listed in Table IV. The first number for the execution time of our method is for deblurring one frame, and the second number is for generating the next frame. Table IV reveals our method needs much less time to train. While the training time of some methods is not available, compared with them, our method has fewer tuning parameters. Although the execution time of the proposed method is slower than that of some of the other methods, our method provides a good trade-off between the quality of the results and the execution time.

F. Comparison with EDI

The EDI model [5] is a non-learning method that makes good use of the event sequence for deblurring and has relatively good performance. However, as mentioned before, it suffers from noise along the edges of the restored image. In addition, some images in the real dataset obtained from [5] cannot be restored well, as shown in Figure 10. There is significant noise in the background and the white blocks of the checkerboard, and the restored edges are not smooth due to unrealistic artifacts. The reason is that the threshold and the bandwidth influence the distribution of events, resulting in poor performance of the EDI model. The EDI method does not consider the correlation of neighboring pixels. In contrast, our network uses weights to integrate events. Therefore, the edges have better photometric consistency than in the EDI model.

G. Failure samples

There are indeed some pictures in the real dataset from [5] which our method cannot restore well, as shown in Figure 11. It is due to that the event camera parameters in our synthetic dataset for training are different from those used in [5]. The steep intensity change like a checkerboard is not covered in our training data, resulting in poor performance as the example shown in Figure 11.

V. CONCLUSION

We propose a residual learning-based deblurring method using event cameras, with a modified U-Net structure with

Fig. 10: Comparison between EDI and ours. (a) Blurry images. (b) Stacked events. (c) and (d) Deblurring results of Pan et al. [62] and ours on the real dataset. The restored images in (c) look unrealistic, whereas those in (d) are more smooth and have better photometric consistency.

DenseNet blocks in each layer. Due to the high temporal resolution of event cameras, motion information is encoded in the output event sequence. The use of events provides advantages over image-based methods, and more accurate results than existing event-based methods are obtained due to the effectiveness of the proposed residual representation and network. The Conv-LSTM blocks facilitate the HFR video generation using the restored image and events; thus, the proposed method shows better results than state-of-the-art image-based temporal interpolation methods.

However, color event cameras exist, the image resolution is much lower (346×260 and below) than that of an RGB camera, limiting the application of event cameras. In our future work, we plan to combine high-resolution RGB cameras with
event cameras, construct a hybrid system to achieve higher resolution for deblurring and HFR video generation tasks, and capture more real-world scenario data to strengthen our work.

ACKNOWLEDGMENT
This work was supported by National Key R&D Program of China (2021ZD0109803) and National Natural Science Foundation of China under Grant No. 62136001, 62088102.

REFERENCES


**Haoyu Chen** received the BE and ME degree from Peking University in 2017 and 2020. He is currently an engineer at Bilibili. His research interests span event-based vision and computational photography.

**Minggui Teng** is currently a Ph.D. student in the National Engineering Research Center of Video Technology (NERCVT), School of Computer Science, Peking University (PKU), China. He received his BS degree from Peking University in 2021. His research interests are focused on neuromorphic camera and image enhancement.

**Boxin Shi** received the BE degree from the Beijing University of Posts and Telecommunications, the ME degree from Peking University, and the PhD degree from the University of Tokyo, in 2007, 2010, and 2013. He is currently a Boya Young Fellow Assistant Professor and Research Professor at Peking University, where he leads the Camera Intelligence Lab. Before joining PKU, he did postdoctoral research with MIT Media Lab, Singapore University of Technology and Design, Nanyang Technological University from 2013 to 2016, and worked as a researcher in the National Institute of Advanced Industrial Science and Technology from 2016 to 2017. His papers were awarded as Best Paper Runner-Up at International Conference on Computational Photography 2015 and selected as Best Papers from ICCV 2015 for ICCV Special Issue. He has served as an editorial board member of IJCV and an area chair of CVPR/IJCV. He is a senior member of IEEE.

**Yizhou Wang** is a BoYa Professor of School of Computer Science and the vice director of the Center on Frontiers of Computing Studies at Peking University. He received his Bachelors degree in Electrical Engineering from Tsinghua University in 1996, and his Ph.D. in Computer Science from University of California at Los Angeles (UCLA) in 2005. He joined Xerox Palo Alto Research Center (Xerox PARC) as a research staff from 2005 to 2007. He was granted the National Natural Science Fund (NSFC) for Distinguished Young Scholars. Dr. Wang’s research interests include computational vision, statistical modeling and learning, medical image analysis, and digital visual arts.

**Tiejun Huang** received the B.Sc. and M.Sc. degrees in computer science from Wuhan University of Technology, China in 1992 and 1995, respectively, and the Ph.D. degree in pattern recognition and image analysis from Huazhong (Central China) University of Science and Technology in 1998. He is currently a professor with the School of Computer Science, Peking University, and the Director of the Beijing Academy for Artificial Intelligence. His research areas include visual information processing and neuromorphic computing. He is a Fellow of CAAI, CCF, CSIG and vice chair of the China National General Group on AI Standardization. He published 300+ peer-reviewed papers on leading journals and conferences, and also co-editor of 4 ISO/IEC standards, 5 National standards and 4 IEEE standards. He holds 100+ granted patents. Professor Huang received National Award for Science and Technology of China (Tier-2) for three times (2010, 2012 & 2017).