EvTexture: Event-driven Texture Enhancement for Video Super-Resolution

Dachun Kai [1]   **Jiayao Lu** [1]   **Yueyi Zhang** [1 2]   **Xiaoyan Sun** [1 2]

Abstract

Event-based vision has drawn increasing attention due to its unique characteristics, such as high temporal resolution and high dynamic range. It has been used in video super-resolution (VSR) recently to enhance the flow estimation and temporal alignment. Rather than for motion learning, we propose in this paper the first VSR method that utilizes event signals for texture enhancement. Our method, called EvTexture, leverages high-frequency details of events to better recover texture regions in VSR. In our EvTexture, a new texture enhancement branch is presented. We further introduce an iterative texture enhancement module to progressively explore the high-temporal-resolution event information for texture restoration. This allows for gradual refinement of texture regions across multiple iterations, leading to more accurate and rich high-resolution details. Experimental results show that our EvTexture achieves state-of-the-art performance on four datasets. For the Vid4 dataset with rich textures, our method can get up to 4.67dB gain compared with recent event-based methods. Code: `https://github.com/DachunKai/EvTexture`.

1. Introduction

Video super-resolution (VSR) aims at restoring high-resolution (HR) videos from their low-resolution (LR) counterparts. It has extensive applications in various domains such as surveillance (Zhang et al., 2010), virtual reality (Liu et al., 2020) and video enhancement (Xue et al., 2019). Compared to single image super-resolution, VSR pays more attention to modeling the temporal relationships between frames, as it tries to predict missing details of the current HR frame from other unaligned frames.

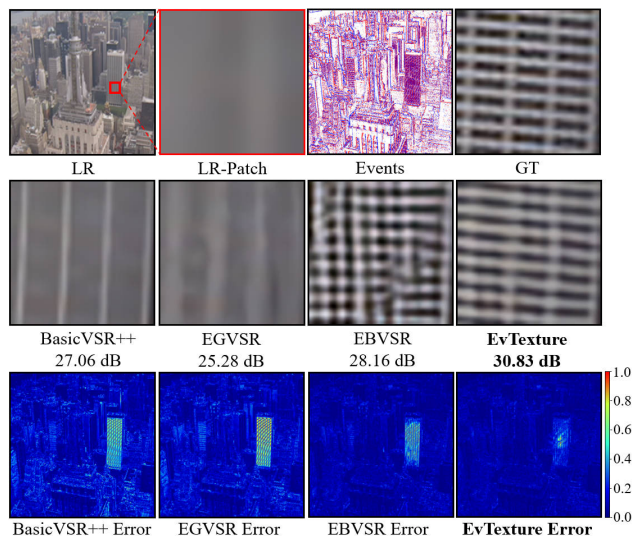Recently, event signals captured by event cameras (Licht-

Figure 1. Comparative results of VSR methods on the City clip in Vid4 (Liu & Sun, 2013). It can be observed that current VSR methods, with (Lu et al., 2023; Kai et al., 2023) or without event signals (Chan et al., 2022), still suffer from blurry textures or jitter effects, resulting in large errors in texture regions. In contrast, our method can predict the texture regions successfully and greatly reduce errors in the restored frames.

steiner et al., 2008) have been used in VSR (Jing et al., 2021; Lu et al., 2023; Kai et al., 2023). Compared with standard cameras, event cameras have very high temporal resolution as well as high dynamic range (Gallego et al., 2020). They thus can provide complementary motion information for VSR. For instance, EGVSR (Lu et al., 2023) introduces a temporal filter branch to explore the motion information from events, such as edges and corners. EBVSR (Kai et al., 2023) utilizes events to enhance the flow estimation and temporal alignment in VSR. However, as shown in Fig. 1, these methods still suffer from large errors in texture regions.

Texture restoration is a very challenging problem in VSR. It is difficult to restore HR textural details from the corresponding LR ones. We notice that there are some attempts working on texture enhancement for image super-resolution (Cai et al., 2022), while for VSR, most methods focus on addressing issues caused by large motion (Wang et al., 2019; Lin et al., 2022) or occlusions (Chan et al., 2022). To our best knowledge, little work has been presented on texture restoration in VSR so far.
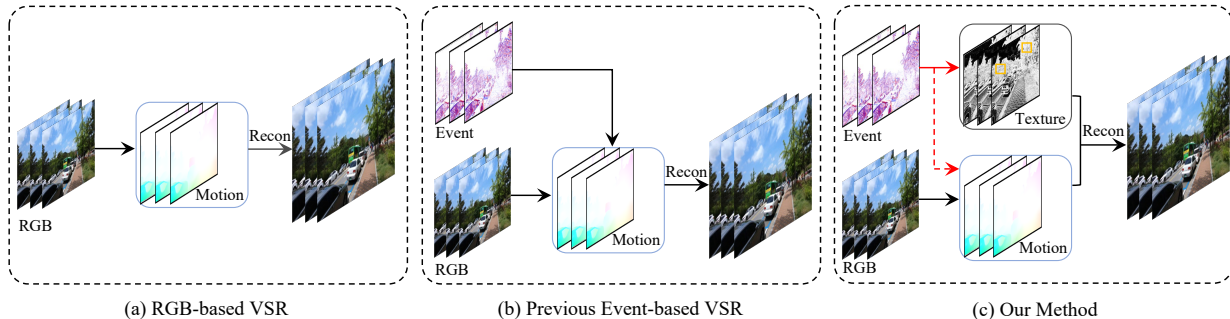
[1]University of Science and Technology of China [2]Institute of Artificial Intelligence, Hefei Comprehensive National Science Center. Correspondence to: Yueyi Zhang <zhyuey@ustc.edu.cn>.

Figure 2. Comparisons of VSR learning process. RGB-based methods (Chan et al., 2022; Lin et al., 2022) usually focus on motion leaning to recover the missing details from other unaligned frames. Previous event-based methods (Lu et al., 2023; Kai et al., 2023) have attempted to use events to enhance the motion learning. In contrast, our method is the first to utilize events to enhance the texture restoration in VSR. The red dotted line is an optional branch, where our method can easily adapt to approaches that use events to enhance the motion learning.

We observe that event signals are not only with high temporal resolution but also full of high-frequency dynamic details, which are desirable for texture restoration in VSR. We thus propose utilizing the event signal for texture restoration and present an event-driven texture enhancement neural network, EvTexture, for VSR. Unlike other event-related methods that use event signals directly in HR frame estimation, our EvTexture progressively recovers the high-frequency textural information in two ways. Firstly, we present a two-branch structure in which the texture enhancement branch is introduced in addition to the motion branch to enhance the texture details. Secondly, we present an Iterative Texture Enhancement (ITE) module to progressively explore the high-temporal-resolution event information for texture restoration. This approach enables a step-by-step enhancement of texture regions across multiple iterations, resulting in more accurate and rich HR details. Experimental results on four datasets demonstrate the effectiveness of our proposed EvTexture. Our EvTexture significantly enhances the performance of VSR especially in texture regions.

The main contributions of this paper are as follows:

- We propose the first event-driven scheme for texture restoration in VSR.
- We propose recovering high-frequency textural information progressively by our presented texture enhancement branch coupled with an ITE module.
- Our proposed texture restoration method achieves state-of-the-art performance on four VSR benchmarks and especially excels in restoring texture-rich clips.

2. Related Work

2.1. Video Super-Resolution

Based on whether the input data uses solely RGB frames or includes additional event signals, VSR methods can be categorized as RGB-based VSR or event-based VSR.

RGB-based VSR. Previous RGB-based VSR methods primarily exploit temporal redundancy between neighboring frames, emphasizing motion learning (Fig. 2(a)). Techniques like optical flow estimation (Xiao et al., 2021; Chan et al., 2021; 2022) and deformable convolutions (Wang et al., 2019; Liang et al., 2022) are used to create advanced alignment modules. Typically, Lin et al. (2022) proposed an unsupervised flow-aligned sequence-to-sequence model (S2SVR) to model the inter-frame relation. However, while these methods get good results in large motion regions, they struggle for hard cases with complex textures, as high-frequency information is lost in LR videos.

Event-based VSR. Event cameras, also known as neuromorphic cameras, are a new type of visual sensor that operates on the principle of asynchronously capturing brightness changes (Gallego et al., 2020). Incorporating event signals into VSR tasks has gained great attention (Jing et al., 2021; Lu et al., 2023; Kai et al., 2023). These techniques mainly focus on using events to enhance the motion learning (Fig. 2(b)). Recently, Kai et al. (2023) proposed to estimate nonlinear flow from events to enhance the temporal alignment in VSR. However, they have not fully exploited the high-frequency dynamic details of events to address the significant challenges associated with texture regions.

2.2. Texture Restoration

We study the fundamental issue of texture restoration in VSR. This challenging problem not only needs to recover high-frequency details but also ensures temporal consistency during video playback given the rich details. While there have been efforts in texture region enhancement in single image super-resolution (Ma et al., 2021; Cai et al., 2022), little work has been dedicated to texture restoration in VSR.

2.3. Iterative Refinement

Recently, Teed & Deng (2020) proposed to iteratively update a flow field, using a recurrent GRU-based update operator.

(a) The Overall Framework of **EvTexture**

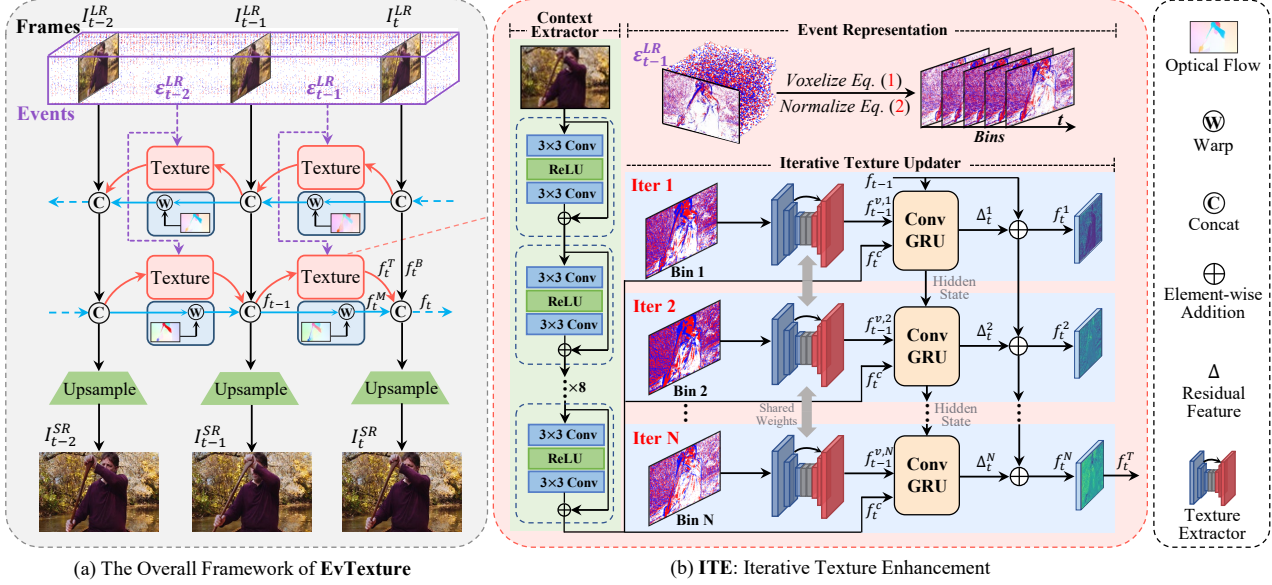(b) **ITE**: Iterative Texture Enhancement

Figure 3. Network architecture of EvTexture. (a) EvTexture adopts a bidirectional recurrent network, where features are propagated forward and backward. At each timestamp, it includes a motion branch and a parallel texture branch to explicitly enhance the restoration of texture regions. (b) In the texture branch, the ITE module plays a key role. It progressively refines the feature across multiple iterations, leveraging high-frequency textural information from events along with context information from the current frame.

This idea has gradually been applied to other tasks as well, such as stereo (Lipson et al., 2021), monocular (Shao et al., 2023), and event-based flow estimation (Wan et al., 2022). For instance, Wan et al. (2022) proposed an iterative update network structure to estimate temporally continuous and spatially dense flow by utilizing frame-event two modalities. In this paper, we take advantage of high-temporal-resolution event signals and employ a GRU-based iterative optimizer to improve the restoration of texture regions in VSR.

3. Preliminary

3.1. Event Representation

The original event stream can be denoted as a set of 4-tuples $\mathcal{E} = \{e_k\}_{k=1}^{N_e}$, where N_e represents the number of events. Each event e_k contains four attributes: (x_k, y_k, t_k, p_k), where (x_k, y_k), t_k and p_k represent the spatial coordinate, timestamp, and polarity of brightness change, respectively.

In practice, it is challenging to find an appropriate representation for an event stream that preserves complete temporal information. In this work, we represent the event stream as a grid-like event voxel grid $\mathcal{V}$ as in the previous work (Zhu et al., 2019), which discretizes the time domain into B time bins. Each time bin in the voxel grid is described as:

$$\mathcal{V}(i) = \sum_k p_k \max\left(0, 1 - \left|(i-1) - \frac{t_k - t_0}{t_{N_e} - t_0}(B-1)\right|\right),$$
(1)

where $i \in \{1, \cdots, B\}$ represents the i-th time bin. In our

experiments, consistent with previous studies (Weng et al., 2021; Wan et al., 2022), we also set $B = 5$. Furthermore, to mitigate the impact of hot pixels, we follow the study (Zhu et al., 2021), and normalize the voxel grid $\mathcal{V}$ as:

$$\hat{\mathcal{V}}(i) = \frac{\min(\mathcal{V}(i), \eta)}{\eta},$$
(2)

where η is the 98th percentile value in the non-zero values of $\mathcal{V}$. In this way, we obtain the normalized voxel grid $\hat{\mathcal{V}} \in \mathbb{R}^{H \times W \times B}$, which contains rich high-frequency textural information ("Events" of Fig. 1) and is easily processed by current deep neural networks.

4. Methodology

We propose a novel neural network, named **EvTexture**, to overcome the challenge of texture restoration in VSR by leveraging high-frequency event signals. The architecture of EvTexture is shown in Fig. 3. The input is an LR image sequence $[I_t^{LR}]_{t=1}^T$ consisting of T frames, and the inter-frame events of the $T-1$ intervals $[\mathcal{E}_t^{LR}]_{t=1}^{T-1}$. The output is the corresponding super-resolved image sequence $[I_t^{SR}]_{t=1}^T$.

Our EvTexture adopts a bidirectional recurrent structure, based on BasicVSR (Chan et al., 2021), where features are propagated forward and backward, and propagation modules are interconnected. At each timestamp, it employs a two-branch structure: a motion learning branch and a parallel texture enhancement branch. The former utilizes optical flow to align frames, while the latter leverages events to en-

hance the texture details. Then, features from both branches are fused and propagated to the next timestamp. Finally, the output features are upsampled through pixel shuffle (Shi et al., 2016) layers to reconstruct the HR frames.

4.1. Two-branch Structure

We illustrate the feature learning process in two branches using forward propagation as an example. The only difference in backward propagation is the direction of data flow.

Motion Learning Branch. Within the motion learning branch (the blue box of Fig. 3(a)), following BasicVSR, we use a lightweight optical flow estimation network $\mathcal{S}$ (SpyNet) (Ranjan & Black, 2017) to estimate the optical flow and use it to align the propagation feature. Specifically, at a given timestamp t, we first take I_t^{LR} and I_{t-1}^{LR} as inputs for $\mathcal{S}$, and estimate the flow $F_{t \to t-1}$. We then use this flow to warp the feature f_{t-1} to the current timestamp t, obtaining the feature f_t^M. This process can be expressed as:

$$F_{t \to t-1} = \mathcal{S}\left(I_t^{LR}, I_{t-1}^{LR}\right), \; f_t^M = \mathcal{W}\left(f_{t-1}, F_{t \to t-1}\right), \tag{3}$$

where $\mathcal{W}(\cdot)$ denotes the spatial warping operation. The motion branch provides basic features that can restore most of the simple and smooth regions.

Texture Enhancement Branch. Due to the rich high-frequency details in texture regions, a flow-based alignment method may not handle them adequately, and in some cases, it can even be detrimental (Shi et al., 2022). To address this challenge, we introduce an additional texture enhancement branch to enhance the texture regions explicitly, with the aid of event signals. Specifically, given the event stream $\mathcal{E}_{t-1}^{LR}$ between I_t^{LR} and I_{t-1}^{LR}, the texture branch propagates f_{t-1} and get the texture-enhanced feature f_t^T as:

$$f_t^T = \mathcal{A}\left(f_{t-1}, \mathcal{E}_{t-1}^{LR}, I_t^{LR}\right), \tag{4}$$

where $\mathcal{A}(\cdot)$ refers to our novel ITE module that will be introduced in the following subsection. The texture branch greatly enhances the restoration of texture regions by leveraging high-frequency details from event signals.

Our method adopts an iterative optimization design to optimize textures over time, with well-suited voxelizing of event streams into different temporal segments, as depicted in Eqs. (1) and (2). With this design, we progressively transfer texture details from events to the propagation feature, thus enhancing the restoration of complex texture regions.

4.2. Iterative Texture Enhancement

Directly incorporating event signals into the reconstruction of HR frames neglects to explore the potential relationship between the high-temporal-resolution properties of events

and the high-frequency texture regions. In this work, motivated by an iterative structure (Teed & Deng, 2020), we propose the ITE module to better model the temporal relation between voxel bins and enhance the texture regions. The architecture of this module is shown in Fig. 3(b), which comprises two feature extractors and a GRU-based iterative texture updater.

Feature Extractors. We employ two types of feature extractors: the context extractor $\mathcal{C}$ and the texture extractor $\mathcal{T}$. The parameters of these extractors are shared across all iterations. The context extractor consists of eight residual blocks used in (Wang et al., 2018) and is applied to the current frame I_t^{LR} to extract the context feature f_t^c. The texture extractor aims to extract the texture feature $f_{t-1}^{v,i}$ from a voxel bin $\hat{\mathcal{V}}_{t-1}(i)$ during each iteration. It is implemented using a custom five-layer UNet (Ronneberger et al., 2015), inspired by its robust ability to capture spatial-temporal features (Jiang et al., 2018). The feature extraction process can be formulated as:

$$f_t^c = \mathcal{C}\left(I_t^{LR}\right), f_{t-1}^{v,i} = \mathcal{T}\left(\hat{\mathcal{V}}_{t-1}(i)\right). \tag{5}$$

Here, both the context feature f_t^c and texture feature $f_{t-1}^{v,i}$ have a common feature size of $\mathbb{R}^{C \times H \times W}$, matching the shape of the propagation feature f_{t-1}.

Iterative Texture Updater. After extracting context and texture features, we introduce our texture updater, designed to transfer the textural information from each voxel bin and the context information from the current frame to the propagation feature. The texture updater, shared across each iteration, consists of three ConvGRU (Ballas et al., 2016) layers and five residual blocks (Wang et al., 2018), denoted as $\mathcal{G}$ and $\mathcal{R}$, respectively. The propagation feature f_t^i, initialized with f_{t-1}, is updated in a residual manner as:

$$h_t^i = \mathcal{G}\left(h_t^{i-1}, \left[f_t^c, f_{t-1}^{v,i}\right]\right), \\ \Delta_t^i = \mathcal{R}\left(h_t^i\right), \; f_t^i = f_t^{i-1} + \Delta_t^i. \tag{6}$$

Here, h_t^i is the hidden state in each iteration at timestamp t, also initialized with f_{t-1}. The superscript i represents the i-th iteration, where $i \in \{1, \cdots, N\}$. The iteration number N is equal to the number of voxel bins, namely B. $[\cdot, \cdot]$ is the concatenating operation. After N iterations, we obtain the enhanced texture feature f_t^T as:

$$f_t^T = f_{t-1} + \sum_{i=1}^{N} \Delta_t^i, \tag{7}$$

which includes the rich textural details transferred from each voxel bin. Our ITE module efficiently transfers high-frequency textual information from event signals to the propagation features and enables progressive enhancement of textural details across multiple iterations.

Table 1. Quantitative comparison (PSNR↑/SSIM↑) on Vid4 (Liu & Sun, 2013), REDS4 (Nah et al., 2019) and Vimeo-90K-T (Xue et al., 2019) for 4× VSR. All results are calculated on Y-channel except REDS4 (RGB-channel). The input types "I" and "I+E" represent RGB-based and event-based methods, respectively. **Red** and blue colors indicate the best and second-best performances, respectively.

Method	Input Type	Vid4					REDS4	Vimeo-90K-T
		Calendar	City	Foliage	Walk	Average		
EDVR (Wang et al., 2019)	I	23.98/0.8143	27.83/0.8112	26.34/0.7560	31.06/0.9153	27.30/0.8242	31.09/0.8800	37.61/0.9489
BasicVSR (Chan et al., 2021)	I	23.87/0.8094	27.66/0.8050	26.47/0.7710	30.96/0.9148	27.32/0.8265	31.42/0.8909	37.18/0.9450
IconVSR (Chan et al., 2021)	I	24.07/0.8143	27.86/0.8111	26.54/0.7705	31.08/0.9158	27.46/0.8290	31.67/0.8948	37.47/0.9476
RTVAR (Zhou et al., 2022)	I	24.65/0.8270	29.92/0.8428	26.41/0.7652	31.15/0.9167	27.90/0.8380	31.30/0.8850	37.84/0.9498
BasicVSR++ (Chan et al., 2022)	I	24.50/0.8288	28.05/0.8212	26.90/0.7868	31.71/0.9236	27.87/0.8413	32.39/0.9069	37.79/0.9500
RVRT (Liang et al., 2022)	I	24.55/0.8334	28.35/0.8363	26.98/0.7824	31.86/0.9251	27.94/0.8443	32.75/0.9113	38.15/0.9527
VRT (Liang et al., 2024)	I	24.52/0.8296	28.33/0.8308	26.78/0.7754	31.89/0.9258	27.88/0.8404	32.19/0.9006	38.20/0.9530
EGVSR (Lu et al., 2023)	I+E	21.53/0.6932	26.01/0.7068	24.33/0.6651	27.39/0.8574	24.84/0.7330	26.87/0.7790	34.62/0.9185
EBVSR (Kai et al., 2023)	I+E	25.17/0.8548	29.30/0.8846	27.31/0.8187	31.91/0.9265	28.46/0.8701	31.47/0.8919	37.56/0.9490
EvTexture	I+E	26.10/0.8756	31.24/0.9087	28.12/0.8475	32.67/0.9366	29.51/0.8909	32.79/0.9174	38.23/0.9544
EvTexture+	I+E	**26.44/0.8859**	**31.82/0.9217**	**28.21/0.8542**	**32.86/0.9381**	**29.78/0.8983**	**32.93/0.9195**	**38.32/0.9558**

Table 2. Comparison of perceptual similarity (LPIPS↓), parameters and runtime on Vid4 and REDS4 for 4× VSR. The average runtime is computed for a clip containing 10 frames, each with an LR frame size of 180×320, on an NVIDIA V100-16GB GPU.

Method	Vid4	REDS4	#Params (M)	Runtime (ms)
EDVR (Wang et al., 2019)	0.2641	0.2091	20.6	378
BasicVSR (Chan et al., 2021)	0.2783	0.2018	6.3	**63**
IconVSR (Chan et al., 2021)	0.2722	0.1946	8.7	70
BasicVSR++ (Chan et al., 2022)	0.2593	0.1786	7.3	77
RVRT (Liang et al., 2022)	0.2481	0.1728	10.8	183
VRT (Liang et al., 2024)	0.2505	0.1864	35.6	243
EGVSR (Lu et al., 2023)	0.3351	0.3024	**2.6**	193
EBVSR (Kai et al., 2023)	0.2476	0.1996	12.2	92
EvTexture	0.2185	0.1684	8.9	136
EvTexture+	**0.2048**	**0.1642**	10.1	139

Table 3. Quantitative results on CED (Scheerlinck et al., 2019) for 2× and 4× VSR. Metrics are calculated on the RGB-channel. [†] denotes results are from EGVSR (Lu et al., 2023).

Method	Input Type	2×		4×	
		PSNR	SSIM	PSNR	SSIM
DUF (Jo et al., 2018)[†]	I	31.09	0.9183	24.43	0.8177
SOF (Wang et al., 2020)[†]	I	31.84	0.9226	27.00	0.8050
TDAN (Tian et al., 2020)[†]	I	33.74	0.9398	27.88	0.8231
RBPN (Haris et al., 2019)[†]	I	36.66	0.9754	29.80	0.8975
BasicVSR (Chan et al., 2021)	I	39.57	0.9778	32.93	0.9001
E-VSR (Jing et al., 2021)[†]	I+E	37.32	0.9783	30.15	0.9053
EGVSR (Lu et al., 2023)[†]	I+E	38.69	0.9771	31.12	**0.9211**
EBVSR (Kai et al., 2023)	I+E	40.14	0.9801	33.42	0.9075
EvTexture	I+E	40.52	0.9813	33.68	0.9112
EvTexture+	I+E	**40.57**	**0.9815**	**33.71**	0.9126

4.3. Feature Fusion

After the two-branch feature learning, we obtain the motion feature f_t^M and the texture-enhanced feature f_t^T, respectively. We then employ an effective fusion design to aggregate motion and texture features and generate the propagation feature f_t at timestamp t as:

$$f_t = \mathcal{R}\left(I_t^{LR}, f_t^B, [f_t^M, f_t^T]\right). \quad (8)$$

Here, f_t^B is the feature from the backward branch. Finally, the fused feature f_t is passed through pixel shuffle layers for upsampling. The upsampled feature is then added with the bicubic upsampled result of the input frame I_t^{LR} through element-wise addition to obtain the SR frame I_t^{SR}.

5. Experiments

5.1. Datasets

Synthetic datasets. We use two popular datasets for training: Vimeo-90K (Xue et al., 2019) and REDS (Nah et al., 2019). Specifically, for Vimeo-90K, Vid4 (Liu & Sun, 2013) and Vimeo-90K-T serve as our test sets, assessing in the

Y channel. For REDS, we employ REDS4[1] as our test set and evaluate in the RGB channel. Notably, the Vimeo-90K, Vid4, and REDS datasets lack real event data. We follow the previous studies (Jing et al., 2021; Kai et al., 2023), and use the ESIM (Rebecq et al., 2018) simulator to synthesize the corresponding events from clips. The simulated events are then converted to the voxel grid following Eqs. (1) and (2). Voxels are downsampled through bicubic interpolation, aligning with the frame downsampling method.

Real-world datasets. Following previous event-based VSR studies (Lu et al., 2023; Kai et al., 2023), we use the CED (Scheerlinck et al., 2019) dataset for training and evaluating on real-world scenes. The dataset is captured with a DAVIS346 (Brandli et al., 2014) event camera, which outputs temporally synchronized events and frames at a resolution of 260×346. Following (Jing et al., 2021), we select 11 clips[2] from the total of 84 clips as our test set and use the remainder for training. When calculating the metrics, we exclude boundary 8 pixels and evaluate in the RGB channel.

---

[1]Clips 000, 011, 015, 020 of REDS training set.

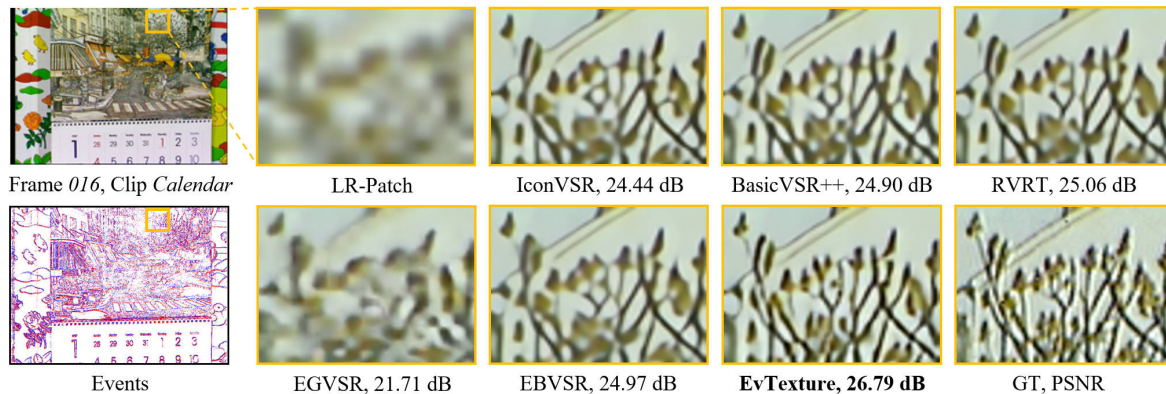[2]Scenes vary from static to dynamic, and indoor to outdoor.

Figure 4. Qualitative comparison on Vid4 (Liu & Sun, 2013). Our method can restore more vivid branches and leaves on the tulip tree.
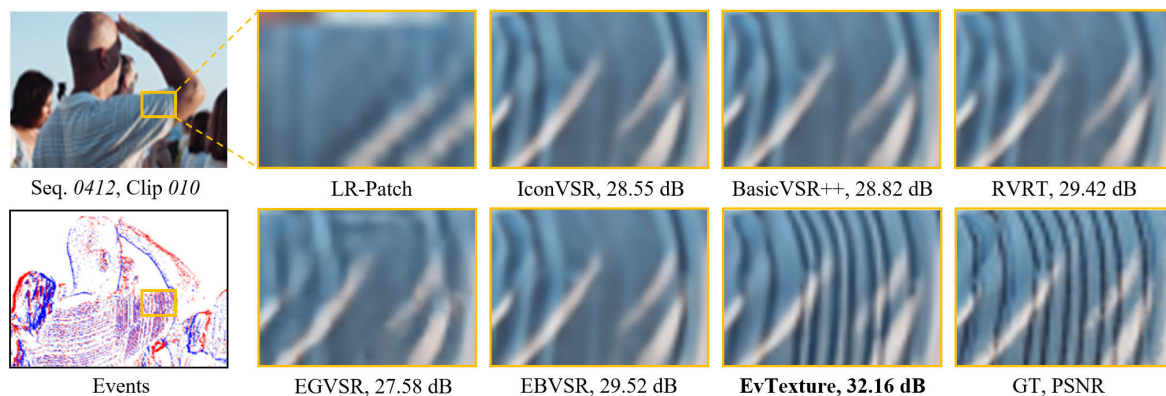

Figure 5. Qualitative comparison on Vimeo-90K-T (Xue et al., 2019). Our method can restore more detailed stripes on clothing surfaces.

5.2. Implementation Details

We use 15 frames as inputs for training and set the mini-batch size as 8 and the input frame size as 64×64. We augment the training data with random horizontal and vertical flips. We train the model for 300K iterations and adopt Adam optimizer (Kingma & Ba, 2015) and Cosine Annealing (Loshchilov & Hutter, 2016) scheduler. The Charbonnier penalty loss (Lai et al., 2017) is applied for supervision. A pre-trained model SpyNet (Ranjan & Black, 2017) is used to estimate optical flow, with other modules trained from scratch. For SpyNet, the initial learning rate is 2.5×10^{-5}, frozen for the first 5K iterations. The initial learning rate for other modules is 2×10^{-4}. The whole training is conducted on 8 NVIDIA RTX3090 GPUs and takes about four days.

5.3. Quantitative Results

We employ two types of baseline methods: RGB-based and event-based. For fair comparisons, we ensure that all methods are trained on the same dataset and evaluated under identical conditions. The evaluation metrics include widely-used PSNR and SSIM. Moreover, considering the common perception-distortion tradeoff (Blau & Michaeli,

2018) problem in restoration algorithms, we additionally evaluate the perceptual similarity metric LPIPS (Zhang et al., 2018), which aligns more closely with human perceptual cognition. We present the comparison results with other SOTA methods on four datasets in Tabs. 1, 2, and 3. These results lead to several important conclusions.

First, our EvTexture utilizes event signals more effectively than other event-based VSR methods. It achieves an impressive performance gain of **+4.67dB** in PSNR over the recent event-based method EGVSR on Vid4. Additionally, our method shows remarkable performance on the real-world CED dataset, achieving +1.83dB gain over EGVSR and +3.20dB improvement over E-VSR. From a perceptual quality perspective, the results in Tab. 2, Figs. 4 and 5 reveal that EvTexture can also offer superior perceptual similarity and restore more vivid and detailed texture regions.

Second, event signals can provide high-frequency information that enhances RGB-based VSR. As shown in Tabs. 1, 2, and 3, EvTexture attains significant improvements over RGB-based methods. For instance, compared to the recent best RGB-based model VRT, EvTexture requires fewer parameters and less runtime and achieves a +1.63dB increase
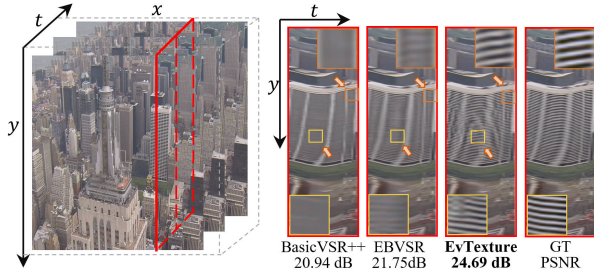
6

Figure 6. Comparison of temporal profile. We select a column (red dotted lines) to observe the changes across time. Our approach creates clearer and more consistent textures over time.

Table 4. Ablation studies of the two-branch structure. The texture branch brings significant improvements on Vid4 and REDS4.

Model ID	Branch		#Params (M)	Vid4	REDS4
	Motion	Texture			
(a)	✓	✗	6.3	27.44/0.8284	31.58/0.8932
(b)	✗	✓	7.5	29.22/0.8814	31.49/0.8942
EvTexture	✓	✓	8.9	**29.51/0.8909**	**32.79/0.9174**

Table 5. Ablation studies about important factors of the Iterative Texture Enhancement module on Vid4.

Model ID	Texture Updater	Iterative Manner	Residual Learning	Iteration Number	#Params (M)	PSNR
(c)	Conv	✓	✓	5	8.8	29.174
(d)	N/A	✗	✓	N/A	7.6	29.087
(e)	ConvGRU	✓	✗	5	8.5	29.158
(f)	ConvGRU	✓	✓	3	8.9	29.463
(g)	ConvGRU	✓	✓	8	8.9	29.324
EvTexture	ConvGRU	✓	✓	5	8.9	**29.507**

on Vid4. Furthermore, our method surpasses the baseline model BasicVSR by +1.37dB on REDS4. Similarly, on CED, EvTexture outperforms BasicVSR by +0.95dB, further verifying the efficacy of incorporating event signals into the VSR framework.

5.4. Qualitative Results

We also perform qualitative comparisons on these datasets. The visual comparisons are shown in Figs. 4 and 5. It is obvious that previous methods, whether using events or not, can not well restore the texture regions, leading to blurry textures or jitter effects. In contrast, our method excels in restoring detailed textures, such as tree branches and stripes on clothing surfaces, resulting in high-quality reconstructions. Sec. F in the appendix provides more visual results, demonstrating the ability of our method to restore high-frequency textural details.

Temporal Consistency. We assess the temporal consistency of our inference results in texture regions using the

Figure 7. Ablations about the two-branch structure on REDS4. Our EvTexture can recover clearer grids on window surfaces.
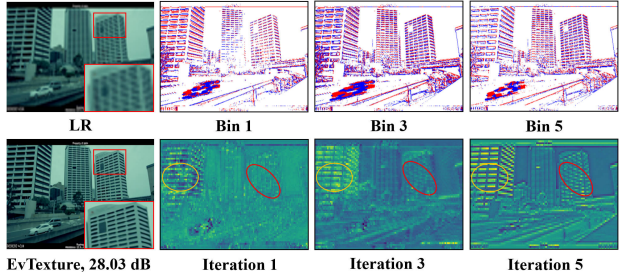

Figure 8. Analysis of iterative texture enhancement. As the iterations advance, the intermediate feature learns clearer texture details from voxel bins, progressively enhancing the restoration quality.

temporal profile (Xiao et al., 2020; Chan et al., 2022). This method visualizes the temporal transition of frames by stacking selected pixel rows from each frame over time. As shown in Fig. 6, our method exhibits superior consistency in texture regions, attributed to our exploration of high-temporal-resolution events. Quantitatively, we achieve a +3.75dB gain over BasicVSR++ and a +2.94dB increase over the recent event-based model EBVSR. Our method not only restores clear textures spatially but also ensures smooth transitions temporally, closely similar to the ground truth.

5.5. Ablation Study

Two-branch Structure. We first conduct ablations of each branch on the Vid4 and REDS4 datasets. The results are shown in Tab. 4. Here, Model (a) only equips the motion learning branch, and Model (b) features the texture enhancement branch. It suggests that on Vid4, Model (b) outperforms Model (a) by a margin of +2.07dB and achieves comparable performance to the EvTexture, indicating the dominant role of the texture branch. Moreover, on REDS4, our proposed model combining both branches achieves the best performance. Fig. 7 shows visual comparisons, where our full model can restore more detailed window grids.

Iterative Texture Enhancement. We also examine some key factors in the ITE module in Tab. 5. Here, in Model (c), we replace the texture updater with three Conv layers. Our EvTexture with the ConvGRU block shows superior perfor-
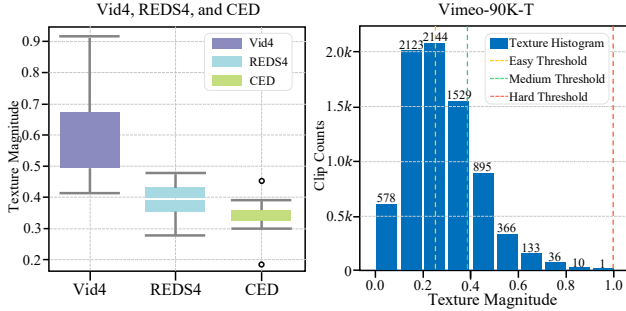
Figure 9. Texture magnitude analysis of four datasets. The Vid4 dataset has the most significant texture. The Vimeo-90K-T dataset exhibits a wide range of texture magnitudes, and we divide it into three difficulty levels (easy, medium, and hard).

Table 6. Quantitative comparison (PSNR↑/SSIM↑) across easy, medium, and hard difficulty levels of Vimeo-90K-T for 4× VSR.

Method	Vimeo-90K-T		
	Easy	Medium	Hard
EDVR (Wang et al., 2019)	41.98/0.975	35.10/0.942	30.40/0.894
BasicVSR (Chan et al., 2021)	41.55/0.973	34.63/0.937	29.97/0.886
IconVSR (Chan et al., 2021)	41.73/0.974	34.86/0.939	30.19/0.890
BasicVSR++ (Chan et al., 2022)	41.98/0.975	35.09/0.941	30.38/0.893
RVRT (Liang et al., 2022)	42.43/0.976	35.69/0.946	30.73/0.902
VRT (Liang et al., 2024)	**42.47**/0.976	35.73/0.947	30.74/0.902
EGVSR (Lu et al., 2023)	38.75/0.959	32.15/0.905	27.90/0.836
EBVSR (Kai et al., 2023)	41.55/0.973	35.09/0.941	30.49/0.896
EvTexture	42.45/**0.977**	**35.83/0.948**	**31.21/0.908**
# of clips	3,907	2,345	1,563
Avg. Texture Mag.	0.16	0.32	0.49

mance, as the gated activation better selects useful information for texture updating. In Model (d), instead of using an iterative update manner, we employ a specialized UNet to directly extract texture features from the whole voxel grid, which leads to a 0.42dB PSNR drop. For Model (e), we remove the residual learning approach used in Eqs. (6) and (7), which causes a 0.35dB PSNR drop. Models (f-g) and our EvTexture have different iteration numbers, and the results indicate that the ITE module with 5-iteration can achieve superior performance. More iterations are not necessary and may lead to worse performance, as there is significant high-frequency information loss in each bin.

Additionally, we analyze the progression of the intermediate feature during iterations in Fig. 8. This analysis further demonstrates that as the texture updater advances, our EvTexture can adaptively learn to extract finer textures with less noise and more clarity from voxel bins for restoration.

6. Discussion

6.1. Effectiveness of Texture Restoration

To further investigate the effectiveness of our method in texture restoration, and inspired by previous image texture

analysis study (Cai et al., 2022), given a video that has T frames of size $H \times W$, we calculate the texture magnitude of a video clip as follows:

$$\frac{\alpha}{T} \sum_{t=1}^{T} \sqrt{\frac{1}{HW} \sum_{i=1}^{H} \sum_{j=1}^{W} \left| I_t(i,j) - \bar{I}_t(i,j) \right|^2}. \quad (9)$$

Here, we first smooth each frame with a Gaussian filter to obtain the blurred image $\bar{I}$, with a kernel size of $(5, 5)$ and $\sigma = 1.5$. Then, we calculate the absolute difference between the original and blurred images to extract high-frequency textural details, and compute the average contrast across the sequence. α is a scaling factor, and in our experiments, we set $\alpha = 10$. The texture magnitude in Eq. (9) ranges from 0 to 1, where a higher value indicates the clip contains richer textures and is more difficult to restore.

Accordingly, we analyze the texture magnitude of the four datasets mentioned in Sec. 5.1, with the results presented in Fig. 9. It reveals that the Vid4 dataset has the most significant texture, followed by REDS4, and then CED. It is worth noting that Vimeo-90K-T has a large amount of 7,815 clips. We categorize these clips into three difficulty levels: easy, medium, and hard, based on each clip's texture magnitude and our careful empirical observations. We then compare our method with SOTA methods across these three difficulty level subsets in Tab. 6.

The results suggest a meaningful conclusion that our EvTexture especially excels in texture-rich datasets and clips. For instance, on two datasets with different texture levels, Vid4 and CED, Tabs. 1 and 3 reveal that EvTexture surpasses the recent event-based method, EBVSR, by +1.05dB and +0.38dB, respectively. Moreover, Tab. 6 shows that our method achieves comparable performance to VRT on the easy subset of Vimeo-90K-T, while our method is more effective on the hard subset, achieving a gain of up to +0.47dB.

6.2. Extension to EvTexture+

Furthermore, despite our primary focus on texture restoration in VSR, we also draw insights from previous event-based flow estimation studies (Wan et al., 2022; Shiba et al., 2022). They exploit the motion information from events to enhance the optical flow, so as to boost the inter-frame alignment like (Kai et al., 2023). Our EvTexture can easily adapt to these approaches. Thus, we extend it to **EvTexture+**, further utilizing events to enhance the motion learning in VSR. Its architecture is detailed in Fig. 12 in the appendix. The results in Tabs. 1, 2 and 3 show that EvTexture+ can bring further performance gains over EvTexture. In some cases, such as on the Vid4 dataset, we notice that the gains from EvTexture+ are minor, which indicates that our EvTexture is already highly effective in texture restoration.

7. Conclusion

This paper presents EvTexture, a novel event-driven texture enhancement network dedicated to texture restoration in VSR by incorporating high-frequency event signals. Our model is based on a recurrent architecture, and we propose a two-branch structure in which a parallel texture enhancement branch is introduced in addition to the motion branch. Furthermore, we propose an iterative texture enhancement module to progressively enhance the texture details through multiple iterations. Experimental results show that our EvTexture outperforms existing SOTA methods and especially excels in restoring finer textures. In the future, we will focus on 1) evaluating our method under more challenging scenarios, including fast-moving and low-light conditions, and 2) extending our approach to handle asymmetrical spatial resolutions between frames and events, which are more common in practical applications.

Acknowledgements

Impact Statement

This paper presents work whose goal is to advance the field of Machine Learning. There are many potential societal consequences of our work, none which we feel must be specifically highlighted here.

References

Ballas, N., Yao, L., Pal, C., and Courville, A. Delving deeper into convolutional networks for learning video representations. In *International Conference on Learning Representations (ICLR)*, 2016.

Blau, Y. and Michaeli, T. The perception-distortion trade-off. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 6228–6237, 2018.

Brandli, C., Muller, L., and Delbruck, T. Real-time, high-speed video decompression using a frame-and event-based DAVIS sensor. In *2014 IEEE International Symposium on Circuits and Systems (ISCAS)*, pp. 686–689. IEEE, 2014.

Cai, Q., Li, J., Li, H., Yang, Y.-H., Wu, F., and Zhang, D. TDPN: Texture and detail-preserving network for single image super-resolution. *IEEE Transactions on Image Processing*, 31:2375–2389, 2022.

Chan, K. C., Wang, X., Yu, K., Dong, C., and Loy, C. C. BasicVSR: The search for essential components in video super-resolution and beyond. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 4947–4956, 2021.

Chan, K. C., Zhou, S., Xu, X., and Loy, C. C. BasicVSR++: Improving video super-resolution with enhanced propagation and alignment. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 5972–5981, 2022.

Gallego, G., Delbrück, T., Orchard, G., Bartolozzi, C., Taba, B., Censi, A., Leutenegger, S., Davison, A. J., Conradt, J., Daniilidis, K., et al. Event-based vision: A survey. *IEEE transactions on pattern analysis and machine intelligence*, 44(1):154–180, 2020.

Haris, M., Shakhnarovich, G., and Ukita, N. Recurrent back-projection network for video super-resolution. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 3897–3906, 2019.

Huang, Z., Zhang, T., Heng, W., Shi, B., and Zhou, S. Real-time intermediate flow estimation for video frame interpolation. In *European Conference on Computer Vision*, pp. 624–642. Springer, 2022.

Jiang, H., Sun, D., Jampani, V., Yang, M.-H., Learned-Miller, E., and Kautz, J. Super slomo: High quality estimation of multiple intermediate frames for video interpolation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 9000–9008, 2018.

Jing, Y., Yang, Y., Wang, X., Song, M., and Tao, D. Turning frequency to resolution: Video super-resolution via event cameras. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 7772–7781, 2021.

Jo, Y., Oh, S. W., Kang, J., and Kim, S. J. Deep video super-resolution network using dynamic upsampling filters without explicit motion compensation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 3224–3232, 2018.

Kai, D., Zhang, Y., and Sun, X. Video Super-Resolution Via Event-Driven Temporal Alignment. In *2023 IEEE International Conference on Image Processing (ICIP)*, pp. 2950–2954. IEEE, 2023.

Kim, T., Chae, Y., Jang, H.-K., and Yoon, K.-J. Event-Based Video Frame Interpolation With Cross-Modal Asymmetric Bidirectional Motion Fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 18032–18042, 2023.

Kingma, D. and Ba, J. Adam: A Method for Stochastic Optimization. In *International Conference on Learning Representations (ICLR)*, 2015.

Lai, W.-S., Huang, J.-B., Ahuja, N., and Yang, M.-H. Deep laplacian pyramid networks for fast and accurate super-resolution. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 624–632, 2017.

Liang, J., Fan, Y., Xiang, X., Ranjan, R., Ilg, E., Green, S., Cao, J., Zhang, K., Timofte, R., and Gool, L. V. Recurrent video restoration transformer with guided deformable attention. *Advances in Neural Information Processing Systems*, 35:378–393, 2022.

Liang, J., Cao, J., Fan, Y., Zhang, K., Ranjan, R., Li, Y., Timofte, R., and Van Gool, L. VRT: A Video Restoration Transformer. *IEEE Transactions on Image Processing*, 33:2171–2182, 2024.

Lichtsteiner, P., Posch, C., and Delbruck, T. A 128×128 120 db 15 μs latency asynchronous temporal contrast vision sensor. *IEEE journal of solid-state circuits*, 43(2): 566–576, 2008.

Lin, J., Hu, X., Cai, Y., Wang, H., Yan, Y., Zou, X., Zhang, Y., and Van Gool, L. Unsupervised flow-aligned sequence-to-sequence learning for video restoration. In *International Conference on Machine Learning*, pp. 13394–13404. PMLR, 2022.

Lipson, L., Teed, Z., and Deng, J. RAFT-stereo: Multilevel recurrent field transforms for stereo matching. In *2021 International Conference on 3D Vision (3DV)*, pp. 218–227. IEEE, 2021.

Liu, C. and Sun, D. On Bayesian adaptive video super resolution. *IEEE transactions on pattern analysis and machine intelligence*, 36(2):346–360, 2013.

Liu, C., Yang, H., Fu, J., and Qian, X. Learning trajectory-aware transformer for video super-resolution. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 5687–5696, 2022.

Liu, H., Ruan, Z., Fang, C., Zhao, P., Shang, F., Liu, Y., and Wang, L. A single frame and multi-frame joint network for 360-degree panorama video super-resolution. *arXiv preprint arXiv:2008.10320*, 2020.

Loshchilov, I. and Hutter, F. SGDR: Stochastic gradient descent with warm restarts. *arXiv preprint arXiv:1608.03983*, 2016.

Lu, Y., Wang, Z., Liu, M., Wang, H., and Wang, L. Learning Spatial-Temporal Implicit Neural Representations for Event-Guided Video Super-Resolution. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 1557–1567, 2023.

Ma, C., Rao, Y., Lu, J., and Zhou, J. Structure-preserving image super-resolution. *IEEE transactions on pattern analysis and machine intelligence*, 44(11):7898–7911, 2021.

Mehta, P., Bukov, M., Wang, C.-H., Day, A. G., Richardson, C., Fisher, C. K., and Schwab, D. J. A high-bias, low-variance introduction to machine learning for physicists. *Physics reports*, 810:1–124, 2019.

Mitrokhin, A., Hua, Z., Fermuller, C., and Aloimonos, Y. Learning visual motion segmentation using event surfaces. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 14414–14423, 2020.

Nah, S., Baik, S., Hong, S., Moon, G., Son, S., Timofte, R., and Mu Lee, K. Ntire 2019 challenge on video deblurring and super-resolution: Dataset and study. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, 2019.

Ranjan, A. and Black, M. J. Optical flow estimation using a spatial pyramid network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 4161–4170, 2017.

Rebecq, H., Gehrig, D., and Scaramuzza, D. ESIM: an open event camera simulator. In *Conference on robot learning*, pp. 969–982. PMLR, 2018.

Ronneberger, O., Fischer, P., and Brox, T. U-Net: Convolutional networks for biomedical image segmentation. In *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2015: 18th International Conference, Munich, Germany, October 5-9, 2015, Proceedings, Part III 18*, pp. 234–241. Springer, 2015.

Scheerlinck, C., Rebecq, H., Stoffregen, T., Barnes, N., Mahony, R., and Scaramuzza, D. CED: Color event camera dataset. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, 2019.

Shao, S., Pei, Z., Wu, X., Liu, Z., Chen, W., and Li, Z. IEBins: Iterative Elastic Bins for Monocular Depth Estimation. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

Shi, S., Gu, J., Xie, L., Wang, X., Yang, Y., and Dong, C. Rethinking alignment in video super-resolution transformers. *Advances in Neural Information Processing Systems*, 35:36081–36093, 2022.

Shi, W., Caballero, J., Huszár, F., Totz, J., Aitken, A. P., Bishop, R., Rueckert, D., and Wang, Z. Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network. In *Proceedings*
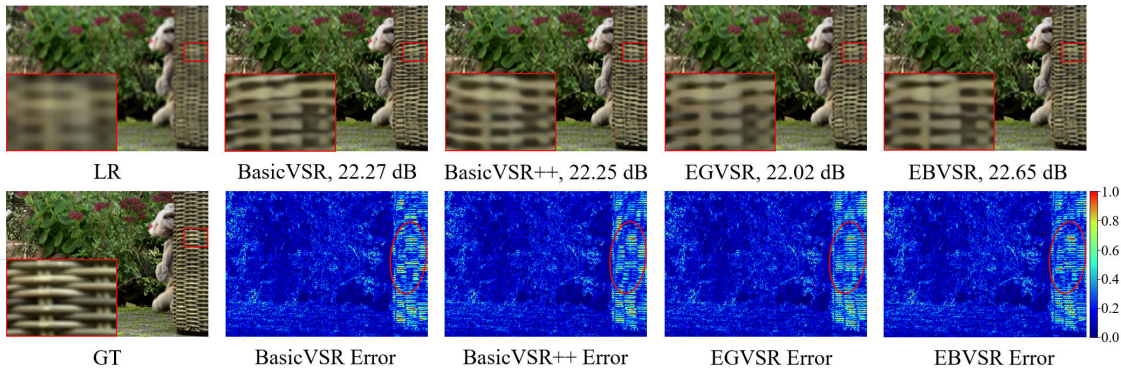
of the IEEE conference on computer vision and pattern recognition, pp. 1874–1883, 2016.

Shiba, S., Aoki, Y., and Gallego, G. Secrets of event-based optical flow. In *European Conference on Computer Vision*, pp. 628–645. Springer, 2022.

Sun, L., Sakaridis, C., Liang, J., Jiang, Q., Yang, K., Sun, P., Ye, Y., Wang, K., and Gool, L. V. Event-based fusion for motion deblurring with cross-modal attention. In *European Conference on Computer Vision*, pp. 412–428. Springer, 2022.

Teed, Z. and Deng, J. RAFT: Recurrent All-pairs Field Transforms for optical flow. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part II 16*, pp. 402–419. Springer, 2020.

Tian, Y., Zhang, Y., Fu, Y., and Xu, C. TDAN: Temporally-deformable alignment network for video super-resolution. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 3360–3369, 2020.

Tulyakov, S., Gehrig, D., Georgoulis, S., Erbach, J., Gehrig, M., Li, Y., and Scaramuzza, D. Time lens: Event-based video frame interpolation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 16155–16164, 2021.

Wan, Z., Dai, Y., and Mao, Y. Learning dense and continuous optical flow from an event camera. *IEEE Transactions on Image Processing*, 31:7237–7251, 2022.

Wang, L., Guo, Y., Liu, L., Lin, Z., Deng, X., and An, W. Deep video super-resolution using HR optical flow estimation. *IEEE Transactions on Image Processing*, 29: 4323–4336, 2020.

Wang, X., Yu, K., Wu, S., Gu, J., Liu, Y., Dong, C., Qiao, Y., and Change Loy, C. ESRGAN: Enhanced super-resolution generative adversarial networks. In *Proceedings of the European conference on computer vision (ECCV) workshops*, 2018.

Wang, X., Chan, K. C., Yu, K., Dong, C., and Change Loy, C. EDVR: Video restoration with enhanced deformable convolutional networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, 2019.

Weng, W., Zhang, Y., and Xiong, Z. Event-based video reconstruction using transformer. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 2563–2572, 2021.

Xiao, Z., Xiong, Z., Fu, X., Liu, D., and Zha, Z.-J. Space-time video super-resolution using temporal profiles. In *Proceedings of the 28th ACM International Conference on Multimedia*, pp. 664–672, 2020.

Xiao, Z., Fu, X., Huang, J., Cheng, Z., and Xiong, Z. Space-time distillation for video super-resolution. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 2113–2122, 2021.

Xue, T., Chen, B., Wu, J., Wei, D., and Freeman, W. T. Video enhancement with task-oriented flow. *International Journal of Computer Vision*, 127:1106–1125, 2019.

Zhang, L., Zhang, H., Shen, H., and Li, P. A super-resolution reconstruction algorithm for surveillance images. *Signal Processing*, 90(3):848–859, 2010.

Zhang, R., Isola, P., Efros, A. A., Shechtman, E., and Wang, O. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 586–595, 2018.

Zhou, K., Li, W., Lu, L., Han, X., and Lu, J. Revisiting temporal alignment for video restoration. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 6053–6062, 2022.

Zhu, A. Z., Yuan, L., Chaney, K., and Daniilidis, K. Unsupervised event-based learning of optical flow, depth, and egomotion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 989–997, 2019.

Zhu, A. Z., Wang, Z., Khant, K., and Daniilidis, K. Event-GAN: Leveraging large scale image datasets for event cameras. In *2021 IEEE International Conference on Computational Photography (ICCP)*, pp. 1–11. IEEE, 2021.
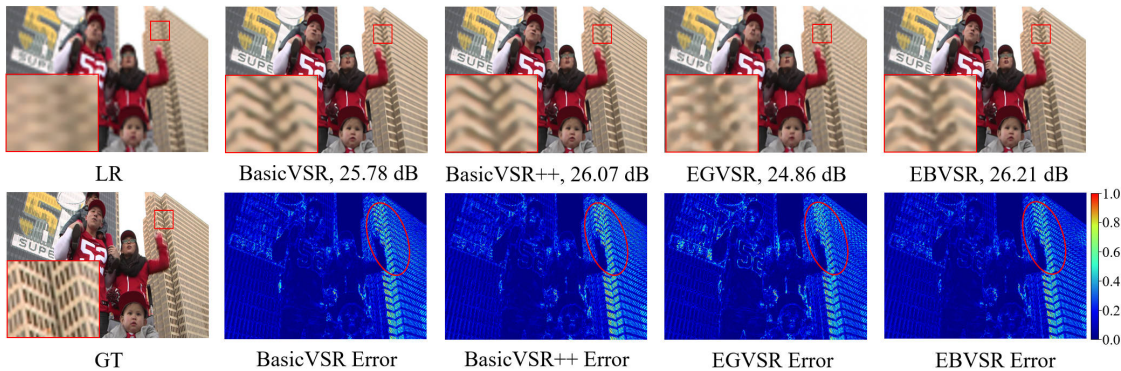
Appendix Contents
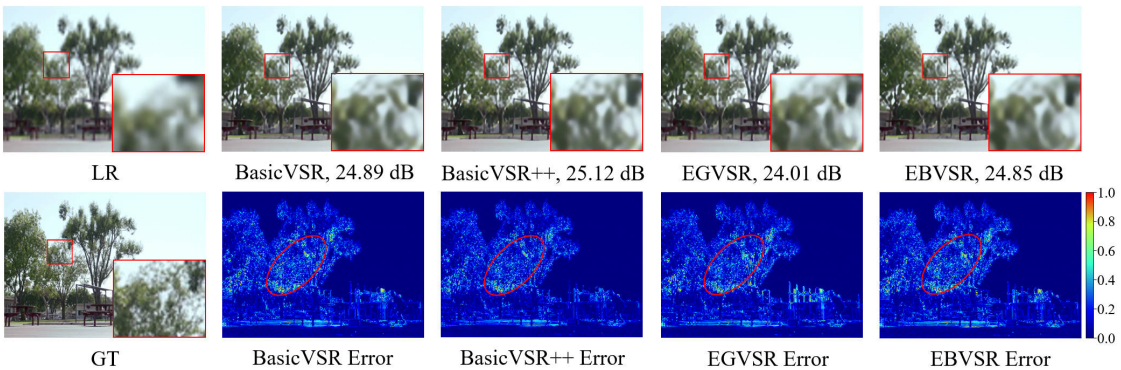
A. Texture Restoration Challenge

Texture restoration is a very challenging problem in VSR. It is hard to predict rich texture details in HR videos from the corresponding LR ones. It is also difficult to preserve the temporal consistency of the texture regions during video playback, given the rich details. As shown in Fig. 10, we provide more examples to illustrate the texture restoration challenge in VSR. We analyze two representative RGB-based methods, BasicVSR (Chan et al., 2021) and BasicVSR++ (Chan et al., 2022), as well as two recent event-based methods, EGVSR (Lu et al., 2023) and EBVSR (Kai et al., 2023). These methods produce blurry textures and exhibit large errors in texture-rich areas such as fabrics, building surfaces, and natural landscapes.

| LR | BasicVSR, 22.27 dB | BasicVSR++, 22.25 dB | EGVSR, 22.02 dB | EBVSR, 22.65 dB |

| GT | BasicVSR Error | BasicVSR++ Error | EGVSR Error | EBVSR Error |

(a) **Significant errors concentrate in the woven basket area.**

| LR | BasicVSR, 25.78 dB | BasicVSR++, 26.07 dB | EGVSR, 24.86 dB | EBVSR, 26.21 dB |

| GT | BasicVSR Error | BasicVSR++ Error | EGVSR Error | EBVSR Error |

(b) **Large errors focus on building surfaces.**

| LR | BasicVSR, 24.89 dB | BasicVSR++, 25.12 dB | EGVSR, 24.01 dB | EBVSR, 24.85 dB |

| GT | BasicVSR Error | BasicVSR++ Error | EGVSR Error | EBVSR Error |

(c) **Considerable errors occur in texture-rich areas, such as tree branches and leaves.**

Figure 10. **Problem analysis.** Existing state-of-the-art VSR methods, such as RGB-based BasicVSR (Chan et al., 2021) and BasicVSR++ (Chan et al., 2022), and event-based EGVSR (Lu et al., 2023) and EBVSR (Kai et al., 2023), still suffer from blurry textures, leading to noticeable errors in texture regions. **Zoomed in for best view.**

B. Motivation about Our Work

(a) Event generation and representation   (b) Comparison of event-based VSR
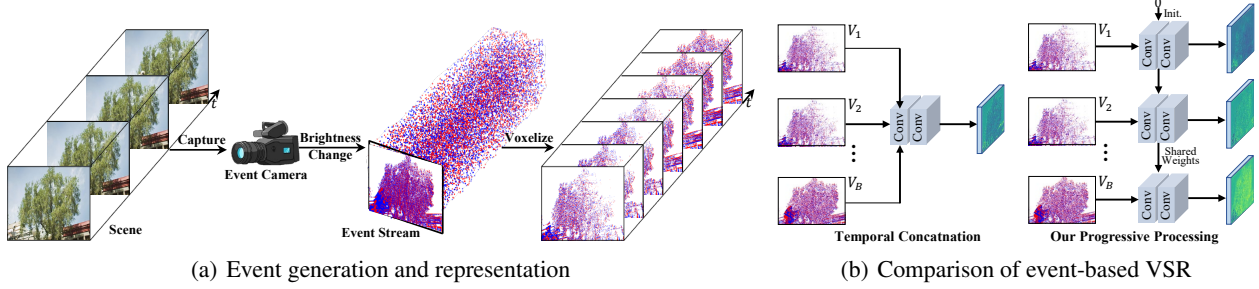
Figure 11. Analysis of event signals and our method. (a) Event cameras asynchronously capture scene brightness changes and output a high-temporal-resolution event stream. We represent these events as voxel grids proposed by Zhu et al. (2019). (b) Previous event-based VSR methods (Jing et al., 2021; Lu et al., 2023; Kai et al., 2023) directly concatenate voxel bins temporally. In contrast, our method processes bins progressively, enhancing textures iteratively and preserving the temporal relationships among these bins.

B.1. Event Generation Model

Event cameras respond to changes in the logarithmic brightness signal $L(u_k, t_k) \doteq \log I(u_k, t_k)$ asynchronously and independently (Gallego et al., 2020). An event is triggered at pixel $u_k = (x_k, y_k)$ and at time t_k as soon as the brightness changes over a pre-setting contrast threshold $\pm C$ $(C > 0)$ as:

$$L(u_k, t_k) - L(u_k, t_k - \Delta t) \geq p_k C, \tag{10}$$

where $p_k \in \{+1, -1\}$ is the polarity of brightness change (increase or decrease), and Δt is the time since the last event at u_k. Consequently, an event is made up of (x_k, y_k, t_k, p_k). The output event stream from an event camera can be denoted as a set of 4-tuples $\mathcal{E} \in \mathbb{R}^{N_e \times 4}$, where N_e represents the number of events.

B.2. High-frequency Textural Information in Events

Fig. 11(a) depicts the event generation and representation model. Events can indeed provide rich high-frequency textural information to enhance VSR results. The reasons are three-fold.

- First, event cameras measure per-pixel brightness changes, and such changes usually occur first at the edges of objects due to object movements. This type of "moving edge" information has been referenced in event-based segmentation tasks (Mitrokhin et al., 2020) and is extensively utilized in event-assisted deblurring studies (Sun et al., 2022).

- Second, unlike standard cameras, which rely on a fixed clock, event cameras asynchronously sample light based on scene dynamics (Gallego et al., 2020). This enables rapid and repeated responses to movements, allowing for multiple samplings in a short period. Thus, events have richer edge information compared to standard frame-based signals.

- Third, it is well-known that edges represent high-frequency information, and rich edges can assist in recognizing textural patterns. As a result, events can provide high-frequency textural information for enhancing VSR.

B.3. Progressive Processing of Events

Fig. 11(b) depicts our approach in utilizing high-temporal-resolution events. Unlike methods that directly concatenate voxel bins over time, we iteratively enhance texture regions by breaking optimization into steps: $t,\ t + \delta,\ t + 2\delta,\ ...\ , t + 1$, where $\delta = 1/B$. This is achieved through our iterative texture updater, which utilizes ConvGRU units. The benefits of this approach are three-fold:

- The iterative structure allows for progressive enhancement of textural details across multiple iterations, resulting in a more precise restoration of the texture regions.

- Instead of directly passing the event data to synthesize texture regions all at once, our structure also models the temporal relationships within events.

- ConvGRU units can utilize information from previous steps (hidden state) to influence current decisions.

C. Extension to EvTexture+

Texture restoration is a very challenging problem in VSR, and our EvTexture is designed to focus on solving this problem with the help of high-frequency textural information from events. Additionally, we also draw insights from previous event-based flow estimation studies (Wan et al., 2022; Shiba et al., 2022) to exploit motion information of events to enhance the flow estimation and temporal alignment. We thus develop the **EvTexture+**, an extension of our EvTexture, further utilizing events to enhance the motion learning in VSR.
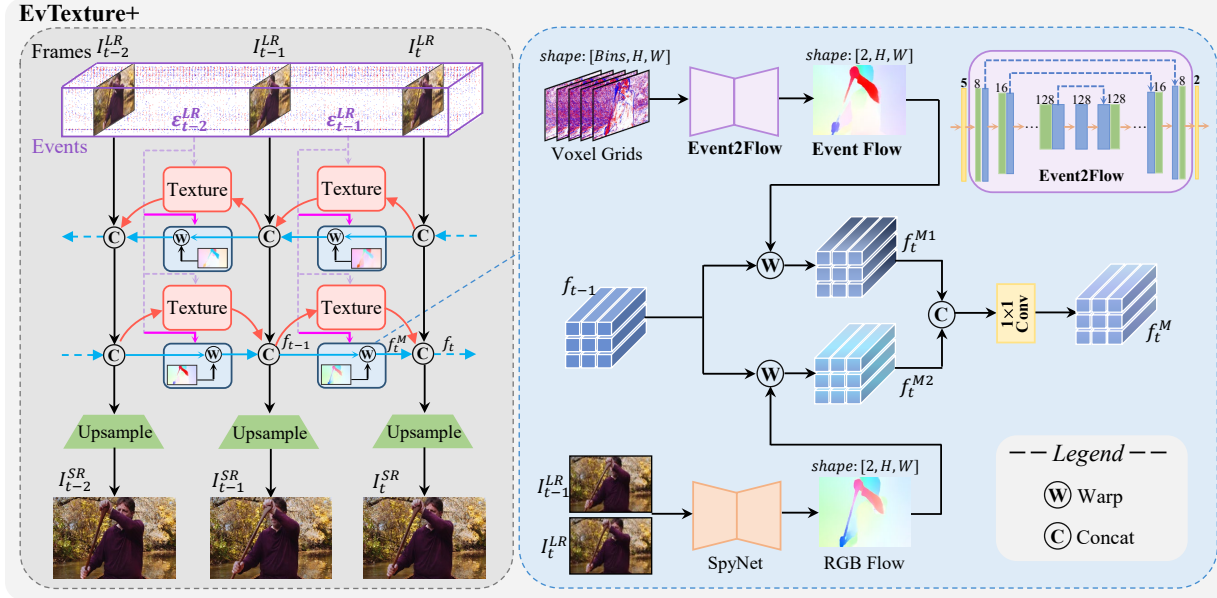

Figure 12. Network architecture of EvTexture+. Compared to EvTexture, it additionally incorporates events to enhance motion learning.

Fig.12 presents the network architecture of **EvTexture+**. The left part of the figure shows that it includes an additional data flow (**magenta** arrow) from events to the motion branch. In the motion branch, events are first converted to the voxel grid as described in Eqs (1) and (2). We design an Event2Flow network to capture nonlinear motion properties in events. This network, similar to EBVSR (Kai et al., 2023), is a custom U-Net (Ronneberger et al., 2015) and generates event-driven optical flow. Afterwards, event flow and RGB flow separately warp f_{t-1} to obtain f_t^{M1} and f_t^{M2}, respectively. f_t^{M1} incorporates rich nonlinear motion information from events, while f_t^{M2} includes color and content information from frames. These two features are fused by a 1×1 Conv to create the motion-enhanced feature f_t^M. It reveals that EvTexture+ can readily adapt to existing event-based VSR methods, which primarily exploit the motion information from events.

The results presented in Tabs. 1, 2, 3 and Sec. D suggest that EvTexture+ achieves additional performance improvements over EvTexture in most cases. In certain cases, for example, on the Vid4 dataset in Tab. 1, the gains from EvTexture+ are minor, which suggests that our EvTexture is already highly effective in texture restoration.

D. Full Experimental Results

In this section, we present detailed clip-by-clip test results for the REDS4 (Nah et al., 2019) and CED (Scheerlinck et al., 2019) datasets, shown in Tabs. 7 and 8. Note that the last column of each table presents the clip's texture magnitude, calculated via Eq. (9). A higher Texture Magnitude (Mag.) value indicates that the clip has richer textures and is more difficult to restore.

Table 7. Clip-by-clip results (PSNR↑/SSIM↑) on REDS4 (Nah et al., 2019) for 4× VSR. Results are evaluated on the RGB-channel. **Red** indicates the best performance. <u>Underlined</u> indicates some significant values that reflect our method's superiority with rich texture clips.

REDS4 Clip Name	RGB-based VSR			Event-based VSR			EvTexture *vs.* VRT	EvTexture *vs.* EBVSR	Texture Mag. (Eq. (9))
	BasicVSR (Chan et al., 2021)	TTVSR (Liu et al., 2022)	VRT (Liang et al., 2024)	EGVSR (Lu et al., 2023)	EBVSR (Kai et al., 2023)	**EvTexture**			
000	28.40/0.8434	28.82/0.8565	28.85/0.8553	25.16/0.7066	28.44/0.8446	**30.72/0.9082**	<u>+1.87/+0.0529</u>	<u>+2.28/+0.0636</u>	<u>0.47</u>
011	32.47/0.8979	33.46/0.9100	33.49/0.9072	26.56/0.7722	32.55/0.8987	**33.72/0.9145**	+0.23/+0.0073	+1.17/+0.0158	0.38
015	34.18/0.9224	35.01/0.9325	**35.26/0.9332**	29.83/0.8526	34.22/0.9235	35.06/0.9314	<u>-0.20/-0.0018</u>	<u>+0.84/+0.0079</u>	<u>0.29</u>
020	30.63/0.9000	31.17/0.9093	31.16/0.9078	25.94/0.7846	30.67/0.9009	**31.65/0.9154**	+0.49/+0.0076	+0.98/+0.0145	0.41
Average	31.42/0.8909	32.12/0.9021	32.19/0.9006	26.87/0.7790	31.47/0.8919	**32.79/0.9174**	+0.60/+0.0168	+1.32/+0.0255	0.39

Table 8. Clip-by-clip results (PSNR↑/SSIM↑) on CED (Scheerlinck et al., 2019) for 2× VSR. Results are evaluated on the RGB-channel. [†] denotes results are from EGVSR (Lu et al., 2023).

CED Clip Name	RGB-based VSR			Event-based VSR			EvTexture *vs.* BasicVSR	EvTexture *vs.* EBVSR	Texture Mag. (Eq. (9))
	TDAN[†] (Tian et al., 2020)	RBPN[†] (Haris et al., 2019)	BasicVSR (Chan et al., 2021)	EGVSR[†] (Lu et al., 2023)	EBVSR (Kai et al., 2023)	**EvTexture**			
people_dynamic_wave	35.83/0.9540	40.07/**0.9868**	39.35/0.9784	38.78/0.9794	39.95/0.9811	**40.39**/0.9824	+1.04/+0.0040	+0.44/+0.0013	0.34
indoors_foosball_2	32.12/0.9339	34.15/0.9739	39.81/0.9766	38.68/0.9750	40.23/0.9780	**40.54/0.9789**	+0.73/+0.0023	+0.31/+0.0009	0.30
simple_wires_2	31.57/0.9466	33.83/0.9739	39.73/0.9832	38.67/0.9815	40.31/0.9849	**40.75/0.9859**	+1.02/+0.0027	+0.44/+0.0010	0.34
people_dynamic_dancing	35.73/0.9566	39.56/**0.9869**	39.60/0.9789	39.06/0.9798	40.22/0.9816	**40.66**/0.9829	+1.06/+0.0040	+0.44/+0.0014	0.33
people_dynamic_jumping	35.42/0.9536	39.44/**0.9859**	39.45/0.9778	38.93/0.9792	40.06/0.9805	**40.45**/0.9819	+1.00/+0.0041	+0.39/+0.0004	0.34
simple_fruit_fast	37.75/0.9440	40.33/0.9782	42.71/0.9815	41.96/0.9831	43.08/0.9830	**43.27/0.9834**	<u>+0.56/+0.0019</u>	<u>+0.19/+0.0004</u>	<u>0.18</u>
outdoor_jumping_infrared_2	28.91/0.9062	30.36/0.9648	39.15/0.9748	38.03/0.9755	39.97/0.9754	**40.53/0.9800**	<u>+1.38/+0.0052</u>	<u>+0.56/+0.0046</u>	<u>0.44</u>
simple_carpet_fast	32.54/0.9006	34.91/0.9502	36.97/0.9672	36.14/0.9635	37.24/0.9689	**37.57/0.9705**	+0.60/+0.0033	+0.33/+0.0016	0.36
people_dynamic_armroll	35.55/0.9541	40.05/**0.9878**	39.35/0.9776	38.84/0.9787	39.95/0.9802	**40.35**/0.9815	+1.00/+0.0039	+0.40/+0.0013	0.34
indoors_kitchen_2	30.67/0.9323	31.51/0.9551	38.45/0.9732	37.68/0.9726	38.88/0.9757	**39.27/0.9769**	+0.82/+0.0037	+0.39/+0.0012	0.33
people_dynamic_sitting	35.09/0.9561	39.03/**0.9862**	39.41/0.9799	38.86/0.9810	40.07/0.9804	**40.54**/0.9837	+1.13/+0.0038	+0.47/+0.0033	0.36
Average	33.74/0.9398	36.66/0.9754	39.57/0.9778	38.69/0.9771	40.14/0.9801	**40.52/0.9813**	+0.95/+0.0035	+0.38/+0.0012	0.33

D.1. Clip-by-clip Results on REDS4

Tab. 7 highlights that our EvTexture performs best on clips '000', '011', and '020' on REDS4. In particular, on the most texture-rich clip, '000', EvTexture outperforms VRT and EBVSR, achieving PSNR improvements of +1.87dB and +2.28dB, respectively. Conversely, on the texture-weak clip '015', our method is 0.20dB lower than VRT and only +0.84dB higher than EBVSR. These results demonstrate our method's superiority, especially in scenes with rich textures.

D.2. Clip-by-clip Results on CED

Tab. 8 suggests our EvTexture achieves the best performance in most scenarios. Notably, on the clip with the richest textures, 'outdoor_jumping_infrared_2', EvTexture surpasses EBVSR by +0.56dB, while on the weaker texture clip, 'simple_fruit_fast', the gain is only +0.19dB. A similar situation also occurs when comparing EvTexture with BasicVSR. The gain of our method is greater on clips with rich textures and less on clips with weaker textures.

E. Performance versus Runtime

Fig. 13 shows plots comparing performance (PSNR), runtime, and number of parameters on four test sets. Our EvTexture outperforms other state-of-the-art methods on these datasets, offering a better balance in terms of performance, parameters, and runtime. On the texture-rich dataset Vid4, our EvTexture and EvTexture+ outperform BasicVSR++ by +1.64dB and +1.91dB, respectively, with only a minimal cost increase in parameters and runtime.
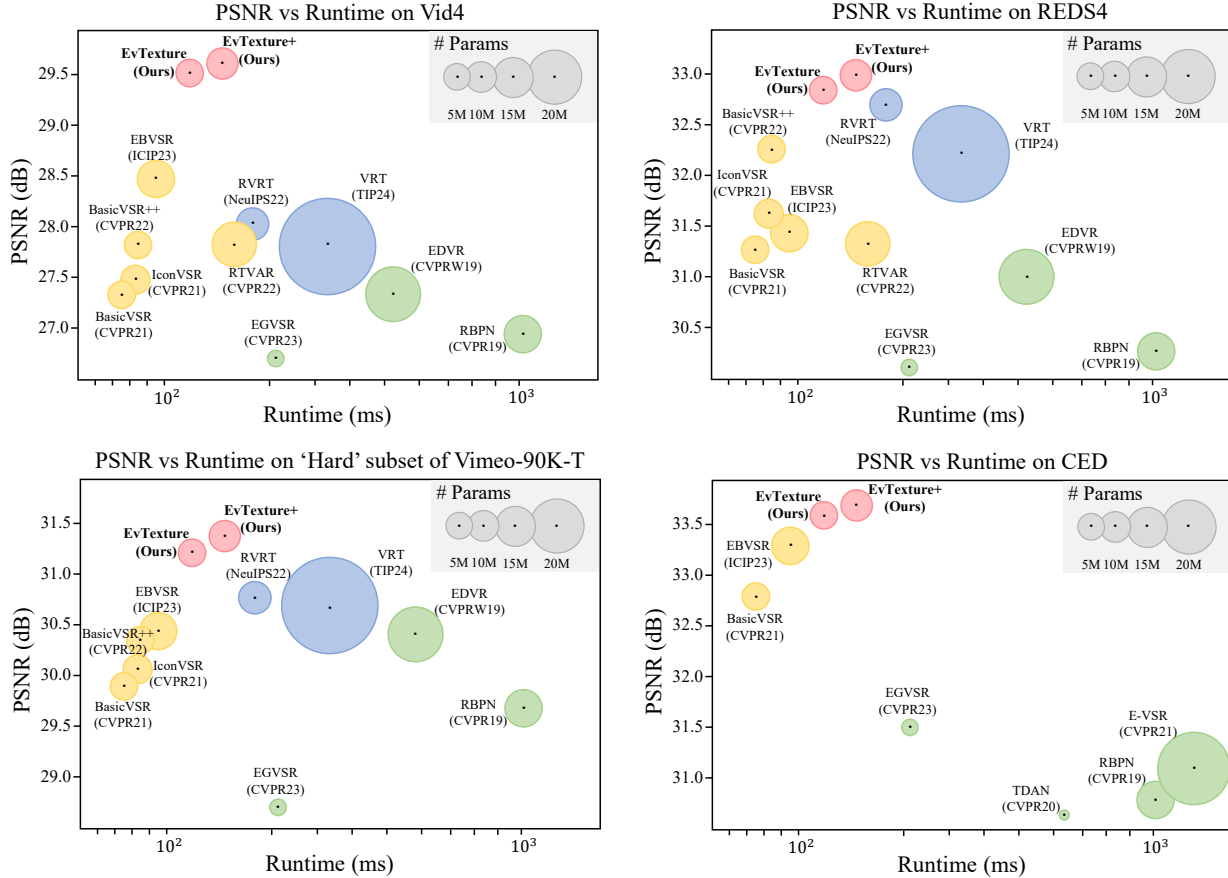

Figure 13. Performance gain on four datasets for $4\times$ VSR. The yellow circles represent RNN-based methods, while the blue ones indicate transformer-based methods. The green circles denote sliding-window approaches, which output only one frame at a time, resulting in longer runtime. Our method is based on an RNN-based VSR approach BasicVSR (Chan et al., 2021).

F. More Visual Results

In this section, to further validate the performance of our EvTexture in restoring texture regions, we provide additional visual comparisons on Vid4 (Liu & Sun, 2013), Vimeo-90K-T (Xue et al., 2019), REDS4 (Nah et al., 2019), and CED (Scheerlinck et al., 2019) datasets. The results are shown in Figs. 14, 15, 16 and 17, respectively. These results show that our EvTexture can successfully restore more vivid and detailed textures across various scenes, including the patterns on woven fabrics, clothes, building surfaces, and the textures of tree branches and leaves.
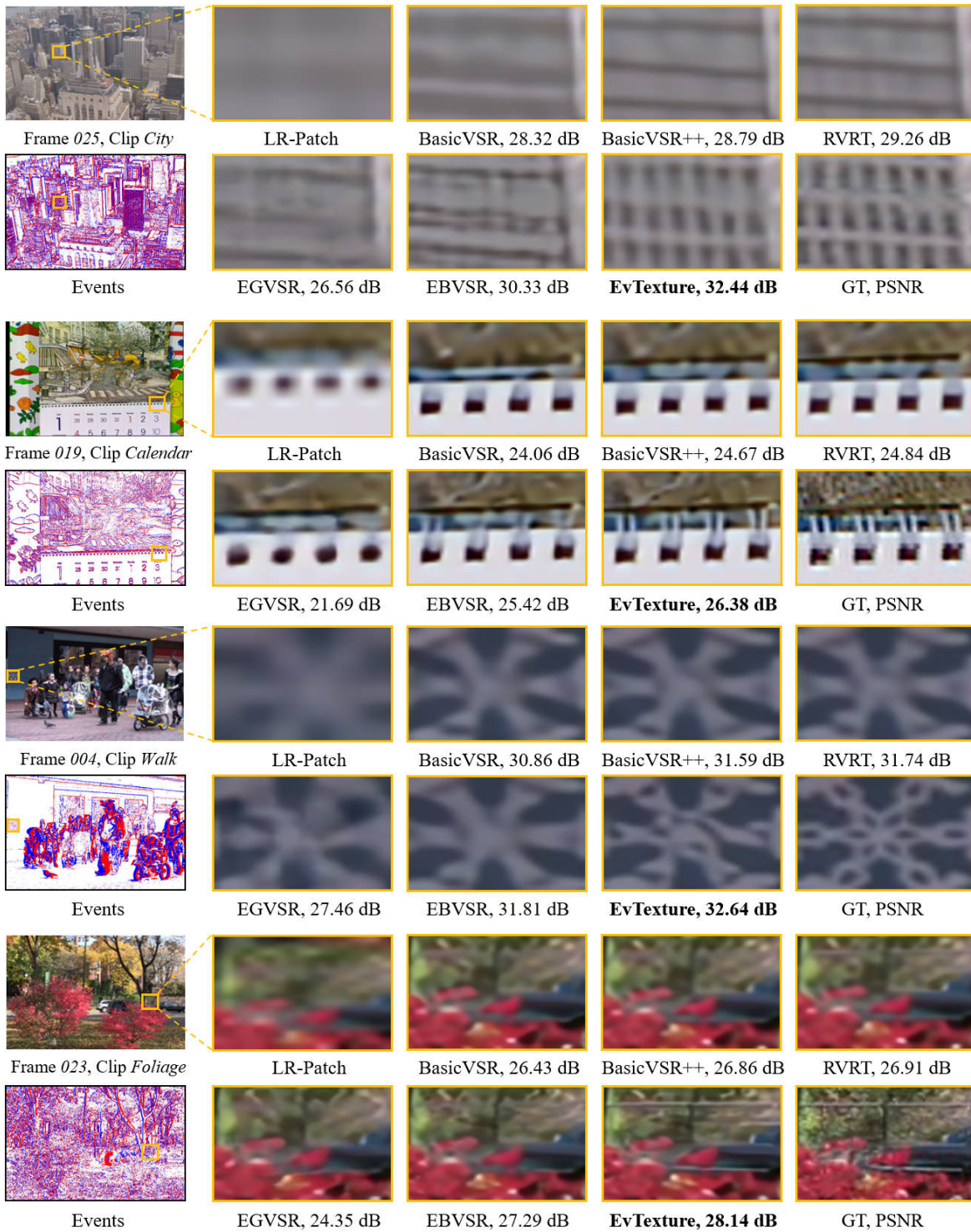
Figure 14. Qualitative comparison on Vid4 (Liu & Sun, 2013) for 4× VSR. **Zoomed in for best view.**
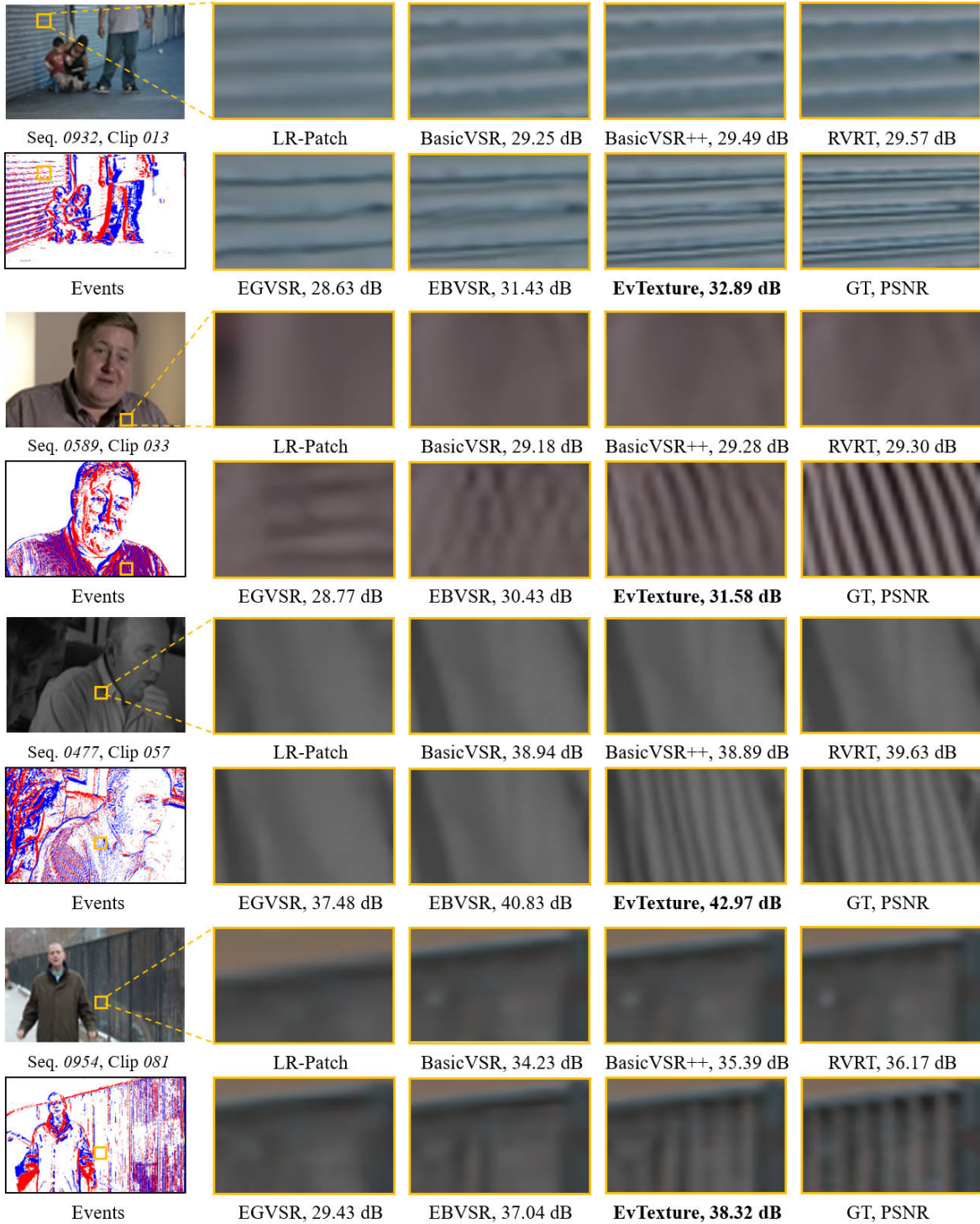
Seq. *0932*, Clip *013*	LR-Patch	BasicVSR, 29.25 dB	BasicVSR++, 29.49 dB	RVRT, 29.57 dB
Events	EGVSR, 28.63 dB	EBVSR, 31.43 dB	**EvTexture, 32.89 dB**	GT, PSNR
Seq. *0589*, Clip *033*	LR-Patch	BasicVSR, 29.18 dB	BasicVSR++, 29.28 dB	RVRT, 29.30 dB
Events	EGVSR, 28.77 dB	EBVSR, 30.43 dB	**EvTexture, 31.58 dB**	GT, PSNR
Seq. *0477*, Clip *057*	LR-Patch	BasicVSR, 38.94 dB	BasicVSR++, 38.89 dB	RVRT, 39.63 dB
Events	EGVSR, 37.48 dB	EBVSR, 40.83 dB	**EvTexture, 42.97 dB**	GT, PSNR
Seq. *0954*, Clip *081*	LR-Patch	BasicVSR, 34.23 dB	BasicVSR++, 35.39 dB	RVRT, 36.17 dB
Events	EGVSR, 29.43 dB	EBVSR, 37.04 dB	**EvTexture, 38.32 dB**	GT, PSNR

Figure 15. Qualitative comparison on Vimeo-90K-T (Xue et al., 2019) 'Hard' for 4× VSR. **Zoomed in for best view.**

Figure 16. Qualitative comparison on REDS4 ([Nah et al.](), 2019) for 4× VSR. **Zoomed in for best view.**
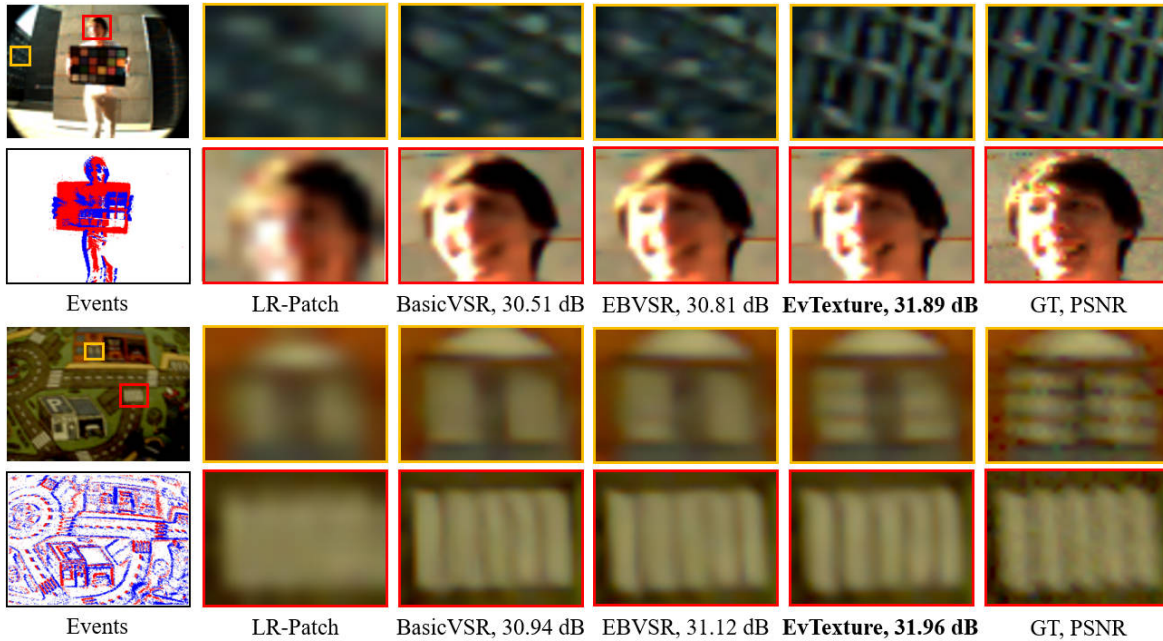

Figure 17. Qualitative comparison on CED ([Scheerlinck et al.](), 2019) for 4× VSR. **Zoomed in for best view.**

G. Dataset Details

G.1. Synthetic Dataset

G.1.1. VID4

The Vid4 (Liu & Sun, 2013) dataset is one of the most popular test datasets for VSR. It consists of four video sequences: 'calendar' (41 frames with a resolution of 576×720), 'city' (34 frames with a resolution of 576×704), 'foliage' (49 frames with a resolution of 480×720), and 'walk' (47 frames with a resolution of 480×720). This dataset provides diverse scenes, making it ideal for assessing VSR algorithms. The results on this dataset are shown in Tabs . 1 and 2.

G.1.2. REDS

The REDS (Nah et al., 2019) dataset, proposed in the NTIRE 2019 Challenge, is a high-quality (720p) video dataset often used for video deblurring and super-resolution tasks. It consists of 270 video sequences, divided into 240 sequences for training and 30 for validation. Each sequence features 100 consecutive frames with a resolution of 720×1280. We follow EDVR (Wang et al., 2019) and select four representative clips, known as REDS4[3], which offers diverse scenes and motions as our test set. The remaining clips are regrouped to form our training dataset, comprising 266 clips. The evaluation results on this dataset are shown in Tabs. 1, 2 and 7.

G.1.3. VIMEO-90K

The Vimeo-90K (Xue et al., 2019) dataset is a large-scale, high-quality video dataset, with 91,701 sequences of 7 frames each at 256×448 resolution. These sequences are extracted from around 39,000 video clips, providing a comprehensive resource for video processing research. The dataset includes 64,612 training clips and 7,824 testing clips, known as Vimeo-90K-T. Following RBPN (Haris et al., 2019), we remove nine clips from Vimeo-90K-T due to their all-black backgrounds.

We then calculate the texture magnitude of the remaining 7,815 clips using Eq. (9). As shown in Fig. 9, Vimeo-90K-T has a wide range of texture magnitudes. Thus, we categorize these clips into three levels: easy, medium, and hard. Specifically, we first sort the clips in ascending order based on their texture magnitudes. After comprehensive user studies and empirical observations, we classify the first 50% (3,907 clips) as easy, the next 30% (2,345 clips) as medium, and the final 20% (1,563 clips) as hard. While our division percentages may not be the most precise, we hope this dataset can provide a better way to study and evaluate texture-related approaches. The test results on three difficulty level subsets are shown in Tab. 6.

G.1.4. SIMULATING EVENTS

Since the Vid4, REDS, and Vimeo-90K datasets lack real event data, we follow the approach widely used in event-based frame interpolation (Tulyakov et al., 2021; Kim et al., 2023), deblurring (Sun et al., 2022) and VSR (Jing et al., 2021; Kai et al., 2023) studies. Accordingly, we use the ESIM (Rebecq et al., 2018) simulator to generate synthetic event data. The ESIM simulator takes a video clip as input and generates corresponding event streams. Before simulating events, we employ the widely used video frame interpolation model RIFE (Huang et al., 2022) to create a high frame rate video with $8\times$ interpolation scale. When simulating events, we apply a threshold c that follows a Gaussian distribution $N(\mu = 1, \sigma = 0.1)$ to accurately mimic the dynamics of real-world scenes.

G.2. Real-world Dataset

G.2.1. CED

Following event-based VSR studies (Jing et al., 2021; Lu et al., 2023; Kai et al., 2023), we use CED (Scheerlinck et al., 2019) as our real-world event dataset. It comprises a collection of color event streams and video sequences, totaling 84 clips in various scenes such as indoor, outdoor, driving, and human moving. We follow the preprocessing steps of E-VSR (Jing et al., 2021) to use 73 clips for training and the remaining 11 clips for testing. The resolution of both frames and events is 260×346. It is important to note that some clips contain a very large amount of frames, for example, 10,541 frames in the 'driving_city_5' clip. Considering memory limitations, processing these frames simultaneously is challenging. Therefore, we process every 15 frames during training, and for testing, we perform inference on every 100 frames. The results on this dataset are shown in Tabs. 3 and 8.

---

[3]Clips 000, 011, 015, 020 of REDS training set.

H. More Discussions

Discussion 1: Why are Vid4 results in Tab. 1 different from original papers?

The results of previous methods in Tab. 1 show slight differences from those reported in their original papers. For instance, the original paper of BasicVSR++ (Chan et al., 2022) reports an average result of 27.79 dB in terms of PSNR on the Vid4 dataset, whereas, in our Tab. 1, we report their result as 27.87 dB. This discrepancy arises because they calculated the average result by treating the four clips of Vid4 as having the same number of frames. However, as mentioned earlier in Sec. G.1.1, **these four clips have different lengths**. Thus, we recalculated their average results, considering the varying clip lengths.

Discussion 2: Is the texture magnitude calculation in Eq. (9) reasonable?

In Fig. 18(a), we present two examples to demonstrate the feasibility of our texture magnitude calculation in Eq. (9). For textures T_1 and T_2 where T_2 is richer, after blurring they become T_1' and T_2'. T_2' quality degrades greatly after blurring, whereas T_1' remains relatively unchanged. Thus, the difference $|T_2 - T_2'|$ is far greater than $|T_1 - T_1'|$. Therefore, the difference map between the original and blurred images reflects texture details. We then calculate the average Root mean square (RMS) contrast over the frames to derive the clip's texture magnitude.

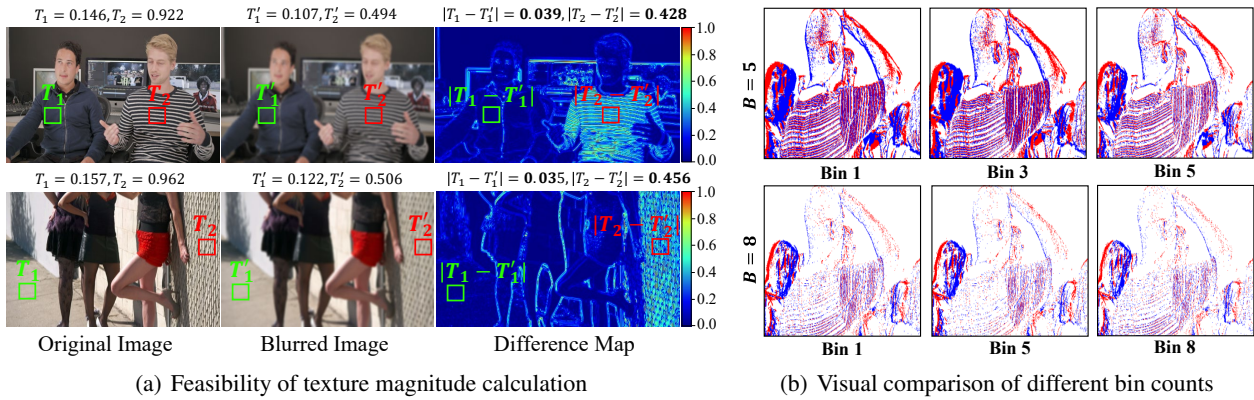| (a) Feasibility of texture magnitude calculation | (b) Visual comparison of different bin counts |

Figure 18. Discussion of texture magnitude calculation and impact of bin counts. (a) The difference value in high-texture areas is significantly larger than in other areas, indicating that using the contrast of the difference map as a measure of texture magnitude is reasonable. (b) More voxel bin counts lead to weak texture information and the evident noise affecting each voxel bin.

Restoring texture regions is a vital and challenging problem in VSR. So far, there are no metrics to measure a clip's texture magnitude in video analysis. Therefore, we draw insights from image texture analysis study (Cai et al., 2022) and provide an auxiliary method in Eq. (9) for analyzing the texture magnitude of clips. This method is easy to use, and the evaluation results verify its feasibility. We hope this can provide a better way to investigate texture-related approaches in the future.

Discussion 3: Why is performance at iteration 8 worse than at 5 in Tab. 5?

In our iterative texture enhancement module, the number of iterations N is equivalent to the number of voxel bins B. As depicted in Fig. 18(b), with $B = 8$, each bin becomes notably sparse. This sparsity, combined with weak texture information and noticeable noise, leads to a decline in performance, particularly in areas with rich textures. This phenomenon is similar to the concept of over-fitting in machine learning (Mehta et al., 2019), where additional iterations are not always beneficial and can even degrade performance.

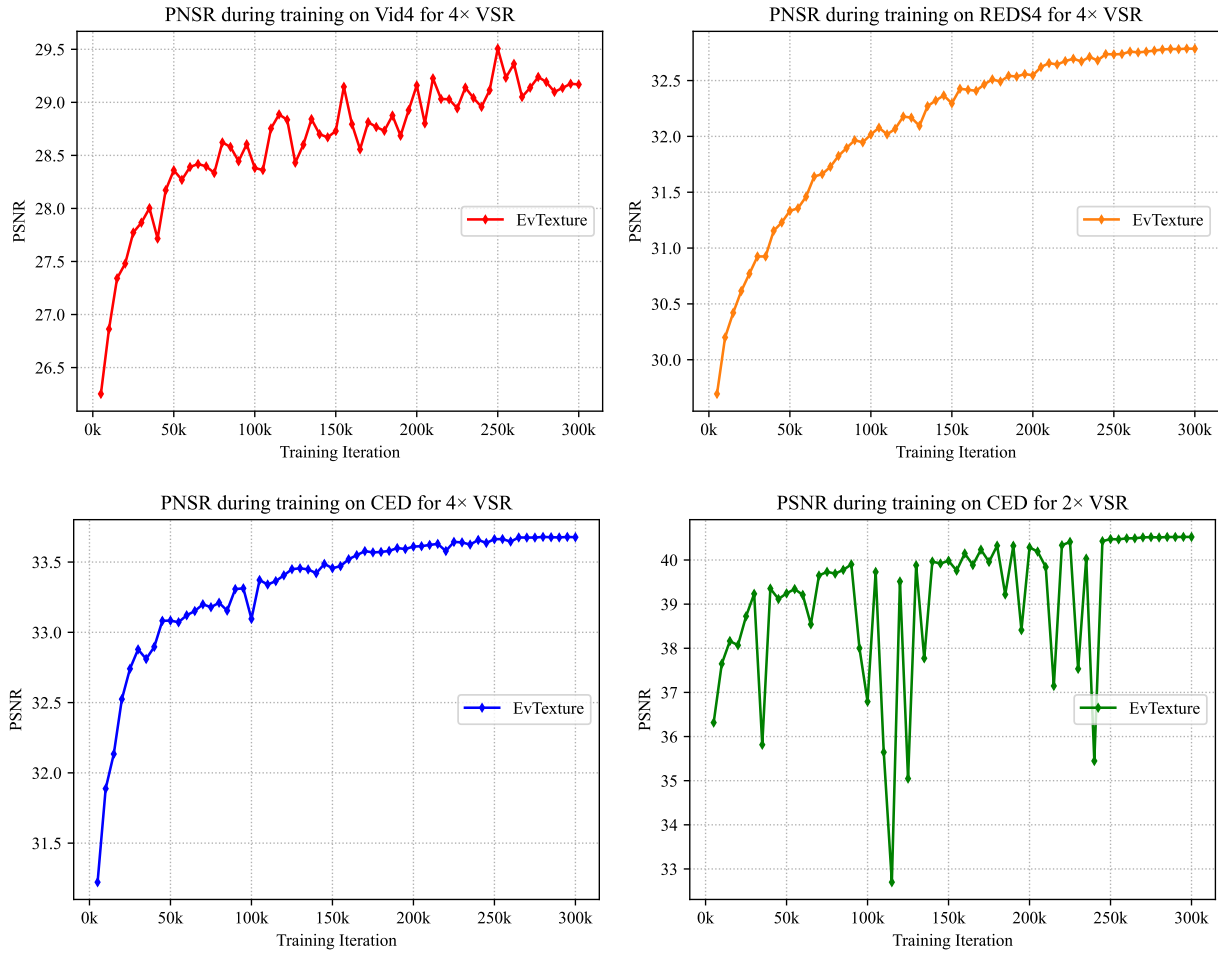# I. Metrics During Training

Figure 19. Metric (PSNR) changes over the iterations of training. These models are all trained from scratch.

Fig. 19 displays plots of PSNR during training, evaluated at saved model checkpoints every 5k iterations over 300k iterations. The curve for 4× scale training exhibits a gradual convergence. Note that the RGB frames from CED (Scheerlinck et al., 2019) are created by demosaicing raw frames and suffer from severe noise, also pointed out by Lu et al. (2023). Additionally, since 2× VSR is a relatively easier task, these factors lead to an oscillating validation curve on CED for 2× VSR.