Detecting Line Segments in Motion-Blurred Images With Events

Huai Yu, Member, IEEE, Hao Li, Senior Member, IEEE, Lei Yu, Member, IEEE, and Gui-Song Xia, Senior Member, IEEE

Abstract—Making line segment detectors more reliable under motion blurs is one of the most important challenges for practical applications, such as visual SLAM and 3D line mapping. Existing line segment detection methods face severe performance degradation for accurately detecting and locating line segments when motion blur occurs. While event data shows strong complementary characteristics to images for minimal blur and edge awareness at high-temporal resolution, potentially beneficial for reliable line segment recognition. To robustly detect line segments over motion blurs, we propose to leverage the complementary information of images and events. Specifically, we first design a general frame-event feature fusion network to extract and fuse the detailed image textures and low-latency event edges, which consists of a channel-attention-based shallow fusion module and a self-attention-based dual hourglass module. We then utilize the state-of-the-art wireframe parsing networks to detect line segments on the fused feature map. Moreover, due to the lack of line segment detection datasets with pairwise motion-blurred images and events, we contribute two datasets, i.e., synthetic FE-Wireframe and realistic FE-Blurframe, for network training and evaluation. Extensive analyses on the component configurations demonstrate the design effectiveness of our fusion network. When compared to the state-of-the-arts, the proposed approach achieves the highest detection accuracy while maintaining comparable real-time performance. In addition to being robust to motion blur, our method also exhibits superior performance for line detection under high dynamic range scenes.

Index Terms—Frame-event fusion, line segment detection, motion blur.

I. INTRODUCTION

LINE segment in imagery encodes rich geometric information and high-level scene layout, which is the basic primitive for image structure perception and 3D geometry estimation.

The codes, datasets, trained models, and demos are released at https://levenberg.github.io/FE-LSD.

Digital Object Identifier 10.1109/TPAMI.2023.3343877

However, the detection of line segments with the emergence of motion blur often suffers from severe performance degradation due to motion ambiguities and texture erasures [1]. As one of the most common artifacts in images, motion blur usually occurs in low illumination conditions and high-speed movements. In contrast to those in images with clear and sharp edges, geometric structures in motion-blurred images are often blurred out, which poses intense challenges to accurately detect and localize line segments, see Fig. 1(a) for an example. This will directly affect the line feature repeatability to establish correspondences over images, then downgrade the performance of subsequent applications, e.g., 3D line mapping [2], visual SLAM [3], and autonomous driving [4], [5].

Although promising progress has been reported in the past decades on line segment detection, e.g., [6], [8], [9], [10], few of them consider the scenarios with high dynamic range and diverse motions, thus often losing their effectiveness when dealing with motion-blurred images. As illustrated in Fig. 1, due to the overlapping of scene information at different times in bluriness, line segments no longer appear as distinctive elongated edges, but as blurred bands mixed with the background. To deal with this issue, an intuitive solution is to first deblur the image and then detect line segments [11]. However, the detection performance highly depends on the quality of motion deblurring, which itself is ill-posed due to motion ambiguities. Moreover, motion deblurring methods are often developed to improve image quality but are not designed for better structure perception [1], [12]. On the other hand, some trails directly detect lines in motion-blurred images [7], [13], [14], which can only determine the distribution or the blurred band of line segments. Nevertheless, it can neither model diverse blurred lines (e.g., rotated around a point on lines) nor geometrically locate the position of line segments. Even utilizing learning-based models [8], [15], training with line annotations is still unable to model the blur distribution and determine line positions. Therefore, accurately detecting and locating line segments solely based on a blurred image are challenging, primarily due to the limited preserved information.

Recently, the newly developed event cameras have attracted increasing attention [16], [17], outputting events of pixel-level brightness change instead of standard intensity frames. In general, event data is generated asynchronously at intensity edges with extremely high temporal resolution (up to μs), thus it is edge-aware and robust to motion blurs. Motivated by these characteristics, we propose to introduce event data into the task of line segment detection in motion-blurred images. On the
A general frame-event feature fusion network is designed to improve the robustness of line detection over diverse camera motions. The designed fusion network on images and events greatly improves the line detection accuracy, especially when encountering motion blurs and high dynamic range scenarios, which will greatly facilitate the robustness of real-world applications such as visual SLAM. To the best of our knowledge, this is the first work that leverages event information for robust line segment detection in motion-blurred images. Our contributions can be summarized as follows:

- Events are introduced to assist motion-blurred images for line segment detection, which can robustly address the performance degradation caused by motion blurs. The idea of fusing images and events fully exploits their complementary properties of low-latency event edges and detailed image textures, thus effectively improving the accuracy and robustness of line detection over diverse camera motions.

- A general frame-event feature fusion network is designed to extract and fuse the information from images and events. The concatenation of shallow fusion and multi-scale decoder fusion fully explores the channel-attention and self-attention mechanism, thus enhancing the feature extraction of events and frames.

- Two frame-event line segment detection datasets, i.e., synthetic FE-Wireframe and real-world FE-Blurframe, are constructed for line segment detection. Extensive qualitative and quantitative results demonstrate the effectiveness of our network design and the superiority of the proposed method when encountering motion blurs and high dynamic range scenarios.

II. RELATED WORK

Line segment detection has been a focused fundamental problem in computer vision over decades, especially with the revolution of deep learning. In this section, we will review the image-based and event-based line segment detection methods. Image-based methods recall the track from traditional edge-based clustering to the latest learning-based algorithms. While for the event-based methods, we mainly analyze the recent adaptions of frame-based methods to events, such as Hough transform [19] and LSD [20].

A. Image-Based Line Segment Detection

Traditional line detection methods mostly rely on edge detection and aggregation of linear geometric properties, such as Hough transform [21], [22], LSD [6], EDTLines [23], and linelet [24]. These methods often follow an unsupervised manner on the CPU and thus can be efficiently deployed on embedding systems. However, their performance highly depends on the edge quality and is sensitive to motion blurs. Recently, learning-based techniques have shown significant progress in detecting line segments. The first learning-based wireframe parsing algorithm DWP [25] uses two CNNs to predict line segments and junctions,
then matches them to obtain the line candidates. Subsequently, AFM [26] is proposed to transform this task into an attraction field regression problem using the line attraction field representation. Those two methods are the first to introduce deep learning to improve the performance of line segment detection. However, they both rely on heuristic post-processing and thus are not end-to-end. To address this issue, a series of end-to-end line segment detection methods [9, 15, 27, 28] have been proposed. L-CNN [15] utilizes a proposal sampling module to conduct the fusion of lines and junctions in the network, eliminating the non-differentiable heuristic post-processing. HAWP [9] combines the attraction field map [26] with the L-CNN network, which effectively improves the quality of line segment proposals. Subsequently, a trainable Hough transform module is added to the existing line segment detection network [29] to enhance the linear geometric constraints. Meanwhile, considering that line segments formulate graphs, PPGNet [27] and LGNN [28] introduce Graph Neural Network (GNN) to fuse the line segments and junctions for optimization, which eventually yields more accurate line detection results. Recently, Transformer was introduced to the line segment detection network. By combining it with CNN, LETR [8] achieves state-of-the-art performance in an end-to-end manner. However, its main drawback is that model training is time-consuming, and inference is relatively slow. Recently, self-supervised methods have also been proposed to overcome the problem of limited training samples [30].

These learning-based line segment detection methods have demonstrated state-of-the-art performance. However, they are trained and tested on clear images without motion blur. For real-world applications such as mobile robotics and autonomous driving, the fast relative motion between the camera and objects will inevitably generate motion blurs in RGB images. The bluriness can significantly compromise or even degenerate the performance of existing methods.

Few existing works are available for line segment detection in motion-blurred images. Debled-Rennesson first presents the concept of blurred line segments in images and gives its mathematical definition [31] by introducing a thickness parameter to describe the line width. Then this blurred line representation model is further extended to gray-scale images, and a semi-automatic line segment selection tool is implemented [32]. FBSD [7] is proposed based on the adaptive directional scanning and the control of assigned thickness to achieve automatic blurred line segment detection. Compared to the traditional LSD [6] and EDLines [23], FBSD demonstrates notable improvements in line location accuracy and recall when dealing with blurs. However, it’s essential to acknowledge that the blurred line definition based on Debled-Rennesson’s approach can only determine the width bands of line segments. Consequently, accurately pinpointing line locations under large motion blurs remains a challenge.

B. Event-Based Line Segment Detection

Over the past few years, several event-based line segment detection methods [5, 33, 34] have been proposed using the newly developed event camera, which does not suffer motion blur due to its asynchronous characteristics and high temporal resolution [16]. The Hough transform is the first to be applied on events [19] for the detection of line segments and has recently been used for the localization and mapping of high-speed trains [34], demonstrating the effectiveness of handling high-speed motions. After that, the ELiSeD [20] is proposed by adapting LSD [6] to the event camera. Based on the assumption of a constant velocity model during a short-time motion period, fast detection and persistent tracking of translating line segments are achieved by plane detection in the event data stream [35]. However, it cannot handle rotating and drastically moving line segments. To improve the robustness over motion speed, an event-based line segment detection method was proposed that uses iteratively weighted least squares fitting [33]. Recently, event cameras have been used for lane detection using basic CNN and transformer networks [5], [36], which give some insights for learning-based line detection. In [37], an online event-based powerline tracker is proposed using hibernation to cope with the line direction changes. The majority of the available methods for detecting events have not been thoroughly evaluated using complex real-world event data. Moreover, there is a lack of publicly available event datasets that can be used as a benchmark, and the performance of line segment detection on real events is yet to be verified.

Although event cameras are robust to motion blurs, the line segment detection performance on pure event stream is unsatisfactory because of the lack of texture information and instability to slow motions or static conditions. Detailed texture can enhance the performance in weakly textured areas and preserve structural integrity. Slow-moving or static event cameras cannot capture informative events. In contrast, RGB cameras are capable of capturing intricate edges and do not suffer from motion blur when moving slowly and being stationary. Therefore, we propose to fully utilize the strengths of both cameras to achieve line segment detection in diverse motion conditions by fusing images and events. However, this task faces many challenges, such as the lack of frame-event fusion strategy for geometric structure perception, and the lack of line segment detection datasets with visual images and event stream data.

III. METHODOLOGY

A. Problem Statement

We first introduce the concept of blurred line segments proposed by Debled-Rennesson [31]. A discretized straight line is defined as:

\[ l = \{(x, y)|c \leq ax - by < c + v\}, \]  
where \((x, y) \in \mathbb{Z}^2\) is the pixel coordinate, \(\{a, b, c, v\} \in \mathbb{Z}^4\) are the four parameters to determine the line segment, \(v\) describes line width. An additional thickness parameter \(\mu\) is defined as the minimum of the horizontal and vertical distances between the lines \(ax - by = c\) and \(ax - by = c + v\), i.e., \(\mu = \min(\max(\mu, v)).\)

In practice, the blurry distribution of line segments is generally complex in motion-blurred images, as shown in Fig. 2. A simple thickness parameter cannot precisely model the line distributions, such as rotated lines. In this paper, line segments
in images are defined as a function of time \( l(t) \), \( t_s \leq t \leq t_s + T \), which are detected on the clear image captured at time \( t \). \( t_s \) can be a start exposure time, and \( T \) is the exposure duration. The imaging process for motion-blurred images can be treated as the average of all clear images \( I(t) \) during an exposure duration \( T \), i.e., \( \bar{I} = \frac{1}{T} \int_{t_s}^{t_s+T} I(t) dt \). Since the temporal resolution is lost by averaging over time, we introduce the high-temporal resolution events \( E = \{(x, y, p_i, t_i)\}_{t_i \in [t_s, t_s+T]} \) into this task. Given the motion-blurred image \( \bar{I} \) and the temporally aligned events \( E \) during \( T \), the line segment detection with the frame and event data can be defined as:

\[
\mathcal{L}(t) = \text{FE-LSD}(\bar{I}, E),
\]

where \( \mathcal{L}(t) \) denotes the set of line segments and FE-LSD denotes the detection function, we mainly focus on the detection results at the exposure ending time \( t = t_s + T \).

### B. Event Representation

Unlike RGB cameras streaming images at a certain frequency, event cameras do not output synchronous frames but event points asynchronously. For each pixel \( u = [x, y]^T \), if the captured luminance \( L \) changes beyond a contrast threshold \( C \) at time \( t \) in the logarithmic domain, an event point \( e = (x, y, t, p) \) is triggered.

\[
\Delta L = p (\log L(x, y, t) - \log L(x, y, t - \Delta t)) \geq C,
\]

where \( \Delta t \) is the time interval between two temporally adjacent events generated at the same pixel. The increase and decrease of luminance will result in positive \((p = +1)\) and negative \((p = -1)\) polarities, respectively.

Event streams have asynchronous and sparse characteristics. To follow the CNN architecture, the asynchronous event streams are often transformed into fixed-size tensors. Recently, the most commonly used event representations include Event Counting [38], Surface of Active Event [39], voxel grid [40], and Event Spike Tensor (EST) [41]. Among these, the EST is a four-dimensional grid of \( H \times W \times B \times 2 \), where the time dimension is uniformly divided into \( B \) bins. Each bin further encodes the polarity as the positive and negative dimensions. Therefore, we select EST as the event representation because it retains the most temporal and polarized information. The EST is calculated by:

\[
\text{EST}_{\pm}(x, y, t) = \sum_{c_i \in E_{\pm}} \delta(x - x_i, y - y_i) \max(0, 1 - |\tau - t_i^c|),
\]

where \( \tau \in \{0, 1, \ldots, B - 1\} \), and \( t_i^c = \frac{B}{T}(t_i - t_0) \). \( t_0 \) is the earliest timestamp of the event stream during time \( T \), \( c_i \in E_{\pm} \) means that the positive and negative EST are computed respectively with the polarities of events. \( \delta(x, y) \) is the 2-dimensional unit-impulse function. When \( x = y = 0 \), \( \delta(x, y) \) equals to 1, otherwise, \( \delta(x, y) = 0 \). This equation is equivalent to the bilinear sampling operation. Fig. 3 shows the event visualization of the raw events, the EST representation, and the overlay of the raw events on the corresponding blurry image. The two spatial-temporal lattices corresponding to the positive and negative polarity of EST are further stacked into an \( H \times W \times 2B \) tensor to fit the common requirement of CNN.

### C. Network Structure

Given the motion-blurred image and the aligned EST data, we need to design an effective fusion strategy to extract the complementary information for line segment detection. Recent advanced line segment detection methods have witnessed the effectiveness of the hourglass network in extracting feature maps [25], [26], thus we consider the fusion of images and events at the feature level with a multi-scale encoder-decoder structure for the two branch inputs. However, if the two input features are directly from the raw images and events, the interaction between the two modalities will be insufficient and the event noises will have negative effects on the later detection of line segments. Therefore, we constructively fuse the shallow and deep features from images and events, and then use two state-of-the-art line segment detectors, i.e., HAWP [9] and ULSD [10] to get the final detection results (Fig. 4). The feature fusion backbone network includes two kinds of modules: (i) Shallow module, which is designed to extract shallow features and suppress the event noises, and (ii) Dual hourglass module, which follows an encoder-decoder structure to conduct a multi-scale feature fusion for the feature map generation.

1) Shallow Module: With the \( H \times W \times 3 \) RGB image and the spatially aligned \( H \times W \times 2B \) EST data, the shallow module first separately down-samples the RGB and EST using the Shallow Layer 1 (including \( 7 \times 7 \) convolution with step size 2, BatchNorm, and ReLU), which can obtain the preliminary RGB and EST features with the same number of channels. Then
the image and event features are fused by the Shallow Fusion Block (SFB), and the fused feature is further added with the original RGB and EST features, respectively. Next, the Shallow Layer 2 (three residual blocks [42] and one maximum pooling module after the first residual block) is used to down-sample and extract features. The refined features are inputted into the second SFB module. Finally, the fused feature is added with the outputs of Shallow Layer 2 to obtain the same dimension image feature \( X_F^{(0)} \) and the event feature \( X_E^{(0)} \), which will be fed into the dual hourglass module subsequently.

The SFB is the core part of the shallow module, and its network structure is shown in Fig. 5. For the input image feature \( X_F \) and the EST feature \( X_E \), the SFB first concatenates them together by channels and retains the number of channels with a 1 × 1 convolution. Then the attention \( Attn_F \) and \( Attn_E \) are computed separately using the channel-attention (CA) blocks [43], which are modified from SENet [44]. The attention is further multiplied with the original features and then added with the features of the other modality to achieve attention-weighted feature fusion. Finally, the fused features are refined using a residual block to obtain the shallow image feature \( X_F^{(i)} \) and event feature \( X_E^{(i)} \). The whole process of the SFB can be described by

\[
X = \text{Conv}_{1 \times 1}(\text{Concat}(X_F, X_E)) \\
Attn = \text{CA}_F(X)
\]

\[
X_F^{(i)} = \text{Res}_F(X_F + X_F \odot Attn_F) \\
X_E^{(i)} = \text{Res}_E(X_E + X_F \odot Attn_E),
\]

where \( \text{Conv}_{1 \times 1} \) denotes 1 × 1 convolution, Concat denotes the concatenation by channel, CA denotes the channel-attention module, Res denotes the residual module, and \( \odot \) denotes the element-wise multiplication.

2) Dual Hourglass Module: As in HAWP [9] and ULSD [10], an encoder-decoder network, i.e., stacked hourglass network [45], is used to obtain the feature map for line segment detection. With the two-branch shallow features, we design the stacked dual hourglass module to further fuse and extract features, as shown in Fig. 4. The image and event features will be first fused by the encoder-decoder network, followed by a residual block. Then the fused feature is added with the input image and event features respectively to recover the two branch features for the next dual hourglass module. For the last dual hourglass module, the fused feature after the residual block is directly outputted as the final feature map for the subsequent line segment detector. The calculation of the stacked dual hourglass module is:

\[
Y^{(i)} = \text{Res}(\text{E-D}(X_F^{(i)}, X_E^{(i)})) \\
X_F^{(i+1)} = X_F^{(i)} + Y^{(i)} \\
X_E^{(i+1)} = X_E^{(i)} + Y^{(i)}
\]

where \( \text{E-D} \) denotes the encoder-decoder network, \( X_F^{(i)} \) and \( X_E^{(i)} \) denote the output image and event features of the \( i \)th dual hourglass module, respectively (\( X_F^{(0)} \) and \( X_E^{(0)} \) are the outputs of the shallow module). \( Y_i \) denotes the fused features of the \( i \)th dual hourglass module output.

The core of the dual hourglass module is the encoder-decoder network, as shown in Fig. 6, which consists of pairwise feature encoders, decoder fusion blocks (DFB), and decoders. Encoders and decoders are implemented by residual blocks. The feature map is down-sampled during feature encoding using a max-pooling layer with step size 2. During decoding, it is up-sampled with step size 2 to ensure the same feature size for the encoder, decoder, and DFB at the same level. The DFB (Fig. 7) consists
Fig. 6. Encoder-decoder network in the dual hourglass module.

Fig. 7. Decoder fusion block in the encoder-decoder network.

of a 1 × 1 convolution and a transformer module using 2 Layer Normalization (LN) modules, a lightweight Multi-head Self-attention (MHSA) module, and an Inverse Residual Feedforward Network (IRFFN) [46]. The DFB module first fuses the input image-event features to one feature by channel concatenation and 1 × 1 convolution, and then uses the transformer [47] to further fuse and extract deep features. The transformer is used mainly for two reasons: (i) Compared with CNN, the transformer has a stronger capacity to capture global features, which is beneficial for the long-distributed target detection such as line segments; (ii) For the fusion of image and event features, the transformer can provide self-attention as well as cross-attention for global cross-modal feature interaction and information fusion.

The structure of the Lightweight MHSA module is shown in the right-bottom of Fig. 7. Compared with the original MHSA [47], this lightweight version adds a k × k convolution (with step size k) to reduce the spatial size of K and V, thus reducing the computational complexity and memory consumption. Additionally, a learnable relative position encoder is used to provide the relative position information between pixels. For the input feature X ∈ ℝ^{h×w×c}, the Lightweight MHSA module computes the output feature Y ∈ ℝ^{h×w×c} as:

\[ X' = \text{Conv}_{k \times k}(X) \]
\[ Q = \text{Conv}_{1 \times 1}(X) \]
\[ K = \text{Conv}_{1 \times 1}(X') \]
\[ V = \text{Conv}_{1 \times 1}(X') \]
\[ \text{Attn} = \text{Softmax} \left( \frac{Q \cdot (K + \text{RPE})^T}{d_h} \right) \]
\[ Y = \text{Attn} \cdot V, \quad (7) \]

where RPE ∈ ℝ^{k \times k \times c} is the relative position encoder and dh is the dimension of each head.

Once the features have gone through the lightweight MHSA module, they are subsequently inputted into the IRFFN [46]. It consists of 1 × 1 convolutions and 3 × 3 deepwise (DW) convolution, where the 3 × 3 deep convolution can reduce the computational cost while extracting local features. The outputs of the first two convolutions are activated by GELU (Gaussian Error Linear Unit) and BatchNorm. The whole IRFFN computation process can be expressed as:

\[ X' = \text{Conv}_{1 \times 1}(X) \]
\[ Y = \text{Conv}_{1 \times 1}(\text{DWConv}(X') + X'), \quad (8) \]

3) Line Segment Detector: Once the fused feature map is obtained, cutting-edge line segment detectors are utilized to execute line segment detection. The detection process is divided into two stages, i.e., the Line Proposal Network (LPN) and the line classification network. The LPN stage involves the junction proposal module and the line proposal module, which produce junction and line proposals, respectively. Subsequently, the line and junction proposals are matched based on their connection relationship to generate the ultimate line segment candidates. Each line candidate is associated with a feature vector from the fused feature map by geometric matching. Then the line classification network is used to classify the line segments by ground truth binary supervision. Finally, candidates with confidence scores that exceed the set threshold are chosen as the final detection outcomes.

By combining the feature fusion backbone with the detector network, line segment detection in motion-blurred images can be achieved using synchronized visual images and EST data. The feature fusion network can be easily applied to existing line segment detectors by simply replacing their feature extraction backbone. To verify the generality of the feature fusion network, we apply it to the current state-of-the-art line segment detectors HAWP [9] and ULSD [10], named FE-HAWP and FE-ULSD respectively.
IV. EXPERIMENTS AND ANALYSIS

In this section, we first detail the dataset, the experimental settings, and evaluation metrics. Then the significant component configurations of FE-LSD are analyzed. Finally, FE-LSD is evaluated and compared with the state-of-the-arts.

A. Dataset

LSD datasets such as Wireframe [25] and YorkUrban [24] play a vital role in LSD methods. However, there is no publicly available LSD dataset with geometrically and temporally aligned RGB images and events. Additionally, it is non-trivial to create precise line segment annotations from scratch on motion-blurred RGB images or EST frames. Based on these requirements, we first build a larger-scale synthetic dataset upon the existing Wireframe dataset [25], i.e., FE-Wireframe. Then, considering there may be large gaps between synthetic and real data, we contribute a real dataset by collecting real data and manually labeling line segments, i.e., FE-Blurframe.

FE-Wireframe: Synthetic data generation aims to avoid laborious line annotation over motion blurs. With the labeled Wireframe dataset [25], we use ESIM [48] to generate the synthetic data and the construction pipeline is shown in Fig. 8. First, the RGB frame \( I_{raw} \) is re-projected to 3D space in the unit coordinate system and then projected to image space to obtain frame \( I_0 \) using the initial camera pose and camera intrinsics. With the simulated camera trajectory over a short time window \( T \), frame \( I_k \) can be generated at any interpolation time \( t_k \). Next, using the two adjacent frames \( I_{k-1} \) and \( I_k \) at time \( t_{k-1} \) and \( t_k \), we can obtain the intensity changes and use (3) to generate the events \( E_k \). The time window \( T = 30 \) ms, and the threshold \( C \) for event contrast is randomly sampled from a uniform distribution \( U(0.05, 0.5) \) for every image sequence. Then we can obtain the event data \( E = \bigcup_{k=0}^{N-1} E_k \) and the simulated frames \( I = \{ I_k \}_{k=0}^{N} \) for \( N + 1 \) images. The number of simulated frames \( N + 1 \) is adaptively sampled based on brightness changes and pixel displacements, which has been proved to be more robust than a constant value in ESIM [48]. It should be noticed that the camera motion trajectory is designed to ensure that the last image \( I_N \) is exactly the raw image \( I_{raw} \) to reuse the existing line annotations. The motion-blurred image \( I_B \) is then obtained by averaging the simulated \( N + 1 \) images. Finally, we obtain the synthetic event data \( E \), the synchronized motion-blurred image \( I_B \), and the line segment annotations. The FE-Wireframe dataset consists of 5000 pairs of frame-event training samples and 462 pairs of testing samples. Several data samples and line annotations are shown in Fig. 9.

FE-Blurframe: For the generation of the real dataset with Frame-Event line segment detection in motion-blur (FE-Blurframe), we build a dual camera system for real-world data collection [49], which consists of a DAVIS 346 event camera and a FLIR RGB camera. The two cameras are triggered by an external signal and we obtain the time offset by maximizing the structural similarity between FLIR images and DAVIS APS images. The two cameras are triggered by an external signal and we obtain the time offset by maximizing the structural similarity between FLIR images and DAVIS APS images. The two cameras are triggered by an external signal and we obtain the time offset by maximizing the structural similarity between FLIR images and DAVIS APS images. The two cameras are triggered by an external signal and we obtain the time offset by maximizing the structural similarity between FLIR images and DAVIS APS images. The two cameras are triggered by an external signal and we obtain the time offset by maximizing the structural similarity between FLIR images and DAVIS APS images. The two cameras are triggered by an external signal and we obtain the time offset by maximizing the structural similarity between FLIR images and DAVIS APS images.

We crop 800 fragments from the 52 sequences, each fragment contains events over 30 ms and the synchronized 7 RGB images. Then, Super SloMo [50] is used to interpolate the RGB images from 7 to 40 frames for natural motion-blurred image generation. 7 frame averaging will generate artifacts (e.g., splines) in blur areas, while interpolated 40 frames are more continuous to produce smooth and natural motion-blurred images. For the annotation, we label line segments on the latest-coming image of the 30-ms interval. Additionally, events are overlayed with this image for annotation validation. Finally, the real dataset contains 800 samples, including clear images, synthetic blurred images, event streams, and line segment annotation. The dataset is randomly divided into a training set with 600 samples and a test set with 200 samples. Several data samples and line annotations are shown in Fig. 10.

B. Experimental Settings and Evaluation Metrics

Experimental setup: In the experiment, the temporal dimension of EST B is set to 5. The sizes of the image and EST are directly re-scaled to 512 × 512 before the feature fusion backbone. In this backbone, the number of DHMs is set as 2, and the number of encoders in each DHM is set as 5. After the feature fusion backbone, we obtain a fused feature map of size 128 × 128 × 256. The parameters and training configurations of FE-HAWP are also set to the same as the original HAWP. The network is trained using the Adam optimizer [51]. The learning rate, weight decay, and training batch size for network training are set to 4 × 10^{-4}, 1 × 10^{-4}, and 4, respectively. In the LPN module, the numbers of junction proposals and line proposals are set as \( K_{junc} = 300 \), \( K_{line} = 5000 \). To train the line classifier, the numbers of positive and negative samples from LPN are set as \( N_{pos} = 300 \), \( N_{neg} = 300 \), respectively. Additionally, we...
add $N_{\text{pos}} = 300$ positive samples from ground truth annotations to the training dataset. To obtain the feature vector for each line, 32 points are uniformly sampled from the line, and the dimension of the feature vector is set as 1024. Meanwhile, the training utilizes a step decay learning strategy, the network is trained by 30 epochs. The learning rate is reduced by a factor of 0.1 after 25 epochs. Since there are two parallel encoders in the dual hourglass module, we use the group convolution to improve efficiency by grouping the two encoders without weight sharing. The data augmentation used in this work includes horizontal and vertical flips and $180^\circ$ rotation. The model training and testing are conducted on a single NVIDIA RTX 3090Ti GPU.

Evaluation metrics: We utilize evaluation metrics that are frequently employed in tasks related to detecting line segments, including $\text{AP}^H$, $\text{F}^H$, mean structural Average Precision (msAP), $\text{sAP}^5$, $\text{sAP}^{10}$, $\text{sAP}^{15}$, junction mean Average Precision (mAP$^J$), and Frames Per Second (FPS).

$\text{AP}^H$ and $\text{F}^H$ are calculated using precision and recall, which follow the traditional LSD [6] and L-CNN [25]. $\text{sAP}$ is proposed in L-CNN [15] to reflect the geometric structure. The distance between the predicted line and the GT line determines whether a predicted line segment is a true positive. $\text{sAP}^5$, $\text{sAP}^{10}$, and $\text{sAP}^{15}$ are the $\text{sAP}$ values when the distance is less than 5, 10, and 15 pixels, respectively. While msAP is the average of $\text{sAP}^5$, $\text{sAP}^{10}$, and $\text{sAP}^{15}$.

$m\text{AP}^J$ is used to measure the precision of junction predictions, which is computed similarly to the $\text{sAP}$ of the line segments. True positive junctions are determined by calculating the euclidean distance between the predicted and GT junctions. $m\text{AP}^J$ is the average accuracy when setting the distance thresholds as 0.5, 1, and 2 pixels.

C. Configuration Analyses

In this section, we first conduct experiments on the synthetic FE-Wireframe dataset to analyze the influence of the three main configurations, i.e., input data fusion, event representation, and fusion strategy. Then, the model trained on the FE-Wireframe...
dataset is fine-tuned on the real dataset to show the transfer learning results.

1) Input Data Fusion: To evaluate the impact of input data on network performance, we train the original HAWP and ULSD models using three different input sets: blurred images only, EST only, and the concatenation of both. It should be noted that the results on blurred images are retrained using blurred images. The results are shown in Table I. As for single-modal input, higher accuracy can be obtained using only image frames. This is because the event data loses a lot of texture information, and event noise hampers junction prediction. Most importantly, when using events to enhance image-based line segment detection, the fusion using naive concatenation can improve the line detection performance. The msAP of HAWP on frame-event concatenation is 3.2 and 20.5 points higher than the model trained with images and events, respectively. The ULSD model shows consistent improvements with 6.1 and 21.9 points higher msAP. These results demonstrate the benefits of edge-aware events in detecting line segments. The detection accuracy can be improved even with a basic concatenation.

2) Event Representation: Different approaches of event representation in event data processing retain varying amounts of information, which can directly impact the performance of FE-LSD. To investigate the impact, four different event representations are tested, i.e., Event Spike Tensor (EST) [41], voxel grid [40], Event Count + Surface of Active Event (EC+SAE) [38], [39], Event Count + Surface of Active Event without polarity (EC+SAE∗). The results are shown in Table II. Among the four representations, when adding polarity information to the Voxel Grid and EC+SAE∗, i.e., EST, and EC+SAE, the comprehensive performance is improved for all metrics. Besides, EST and Voxel Grid retain the temporal information compared with EC+SAE and EC+SAE∗, therefore obtaining higher accuracy which indicates the importance of the temporal information. Therefore, EST is selected as the final event representation of FE-LSD.

3) Fusion Module Analysis: For the fusion of image and event features, we first test the effect of the proposed SFB and DFB, and the results are shown in Table III. SFB is removed directly from the network when it is not used, and element-by-element addition is used to replace DFB when it is unused. We can observe that the network has the lowest accuracy when no fusion module is used. Then, msAP is improved by 2.1 and 1.0 points when the SFB is added, and it increases by 4.6 and 4.4 points when only the DFB is used, for FE-HAWP and FE-ULSD, respectively. When adding SFB to the models with DFB, msAP is improved by 0.3 and 0.9 for FE-HAWP and FE-ULSD. Since different detectors have different capacities, adding SFB to a higher baseline does not always improve performance by a similar margin. The gains brought by DFB are larger than SFB, indicating the importance of the multi-scale encoder-decoder fusion of deep features. The employment of SFB and DFB results in the most accurate results, with FE-HAWP and FE-ULSD achieving 53.2% and 55.2% msAP, respectively. These results verify the effectiveness of the proposed SFB and DFB modules.

Second, we further conduct experiments to discuss the effectiveness of the designed two modules, i.e., the shallow module.
(SM) and the dual hourglass module (DHM). One module is directly removed when testing the other module. The results are shown in Table IV. In general, the DHM plays a more crucial role in detecting line segments than the SM. When combined with the DHM, the SM can slightly enhance the detection performance.

Then, to demonstrate the effectiveness of the shallow module on suppressing the event noises, we show several instances of event images, the corresponding feature maps before and after the shallow module in Fig. 12. For these real events from the FE-Blurframe dataset with “salt noises”, we can clearly observe the “salt noises” in homogeneous areas and burrs around edges. After the shallow module, the feature intensity in the background area is more homogeneous than that before the shallow module. While the feature intensity around image edges is still distinctive compared to that before the shallow module.

Based on the network architecture, we further discuss the number of DHM and the number of feature encoders in DHM in Table V. Without loss of generality, we discuss the numbers within the FE-HAWP framework. For efficiency evaluation, we calculate the average time cost (Time) for extracting the backbone feature map in addition to FPS and the number of parameters (Params). When using the same DHMs, more encoders result in better results. However, the maximal encoder number is 5 because of the down-sampling in the encoder network. When using the same number of encoders, more DHMs result in better accuracy but lower inference speed. However, 24 GB GPU RAM cannot cover the memory needs of 3 DHMs. Therefore, we set the number of DHM as 2, and the number of encoders as 5.

4) Transfer Learning on Real Data: In our experiment, the synthetic data generated on the Wireframe dataset relieves the burden of laborious data annotation on real images and events. However, the generalization ability of the model trained on synthetic data has not been tested on the real-world dataset. Additionally, the synthetic motion-blurred images are different from the real world since 3D information is missing. Therefore, we build a real dataset FE-Blurframe with realistic images and events to discuss the impact of gaps from simulation to reality. To verify the effectiveness of transfer learning, three models, i.e., training on synthetic only (S), training on real only (R), and fine-tuning pre-trained model on real (F), are evaluated on the same real dataset. The results are shown in Table VI.

It can be observed that the models trained on synthetic only obtain 30.7% and 34.4% msAP on the real-world testing set for FE-HAWP and FE-ULSD, respectively. This performance degradation is similar to the HAWP [9] when it is trained on Wireframe only and tested on the YorkUrban dataset. The reasons are mainly two-fold: (i) The significant differences between synthetic and real data; (ii) The synthetic event data is obtained based on the ideal event generation model, while the quality of real event data is affected by the lens, noise, contrast threshold and other factors for event cameras. Then, the performance of the synthetically pre-trained model has been improved substantially after being fine-tuned on the real dataset and also has a significant improvement over the model trained on real only. The msAPs of fusion training are finally improved to 63.3% and 62.9% for the FE-HAWP and FE-ULSD models, respectively. Furthermore, we compare the line detection results on real-world data using HAWP and FE-HAWP in Fig. 13. FE-HAWP obtains consistently better results on diverse motion-blurred images, validating the generalization ability of the proposed method.

Furthermore, we plot the curves of loss and $msAP$ in Fig. 14. The red curves are trained from the pre-trained model, and the green curves are trained from scratch using random initialization. Fine-tuning the pre-trained model can speed up the convergence and improve the final performance. These results indicate a notable difference between the synthetic and the real-world dataset, which seriously affects the model’s generalization ability when evaluating from simulation to the real...
world. Nevertheless, the large-scale synthetic dataset is helpful for model pre-training, thus releasing the data annotation labor and improving the convergence speed and accuracy on the real dataset.

D. Comparisons With the State of the Arts

To evaluate the performance of the proposed FE-LSD, extensive comparisons are conducted on the synthetic FE-Wireframe and real FE-Blurframe datasets, respectively. The competitors include traditional LSD [6], FBSD [7] (designed for blurred images), the learning-based methods L-CNN [15], HAWP [9], and LETR [8]. These learning-based methods are directly evaluated using the officially trained models on the motion-blurred images. Since there is no publicly available source code for the detection methods only using events, we implement LSD and HAWP on the event time surface, HAWP on the event reconstructed images using e2vid method [52] for comparison. For FE-Wireframe, we additionally use the official HAWP and ULSD models on the deblurred images with RED-Net [53], i.e., HAWP† and ULSD†. Finally, since the proposed FE-LSD takes both image and event data inputs, these methods are retrained on concatenating images and EST to give fair play (i.e., L-CNN†, HAWP†, and LETR†). All these methods are tested on a server with an Intel i9 12900 K CPU and a single NVIDIA RTX 3090Ti GPU.

1) Results on the FE-Wireframe Dataset: Fig. 15 shows the qualitative comparisons on the FE-Wireframe dataset. The first row is the input motion-blurred images, and the second is the corresponding event inputs. For better visualization, the detected lines and the manually-labeled lines are plotted on the clear RGB image at the end moment of exposure. Since LSD and FBSD are gradient-based methods, they can detect a large number of segments in motion-blurred images. However, the majority of the line segments are false detection or fragmented segments, resulting in low quantitative metrics. On the other hand, the learning-based methods HAWP and LETR have a low detection rate on blurry images when using the pre-trained models. This is because the officially trained models are driven by clear images and cannot handle the motion-blur issues. However, when we retrain them using the concatenation of blurry images and events, the correct detection is significantly increased. Finally, the proposed FE-HAWP and FE-ULSD have the best performance with the highest detection rate and the lowest false alarm.

Table VII summarizes the quantitative comparisons on the synthetic FE-Wireframe dataset, while Fig. 16 shows the PR curves of sAP$^{10}$ and AP$^{H}$, respectively. When only motion-blurred images are given, traditional LSD, FBSD, and learning-based L-CNN, HAWP, ULSD, and HAWP have low accuracies. The highest msAP obtained by ULSD reaches 5.2%, and FBSD obtains the highest AP$^{H}$ of 24.9%, respectively. When deblurring using the official RED-Net model and detecting line segments using the retrained models, the accuracy is much higher but still worse than the direct retraining on the motion-blurred images. The reason is that the deblurring model lacks generalization ability. Then, we retrain the deblurring model using the clear wireframe training set and detect using the official HAWP and ULSD models, the results are much better. However, they are still incomparable to the results of the proposed FE-HAWP and FE-ULSD. Moreover, the trick of retraining the deblurring model using clear images is impractical for every new dataset. The retrained HAWP on event time surface and reconstructed images have better performance than the official models on blurry images, which demonstrates that events can help detect line segments when encountering fast motions. Then, when taking both image and EST inputs, the retrained models perform much better than only the image input. The highest msAP obtained by the retrained ULSD† achieves

| Method    | sAP$^{10}$ | sAP$^{H}$ | mAP | mAP$^{H}$ | AP$^{10}$ | AP$^{H}$ |
|-----------|-----------|-----------|-----|-----------|-----------|----------|
| LSD [6]   | 0.1       | 0.6       | 1.1 | 0.5       | 3.0       | 24.9     |
| FBSD [7]  | 0.2       | 0.4       | 0.9 | 0.5       | 2.9       | 24.9     |
| L-CNN [15]| 0.3       | 0.9       | 2.8 | 1.1       | 5.2       | 24.9     |
| HAWP [9]  | 3.5       | 6.4       | 12.3| 3.8       | 14.2      | 41.5     |
| ULSD [10]| 5.3       | 11.6      | 22.9| 5.3       | 15.6      | 51.5     |
| L-CNN†    | 34.6      | 40.1      | 45.9| 34.6      | 40.1      | 51.5     |
| HAWP†     | 44.6      | 51.2      | 58.2| 44.6      | 51.2      | 51.5     |
| ULSD†     | 35.5      | 40.1      | 45.9| 35.5      | 40.1      | 51.5     |
| LETR [8]  | 2.8       | 5.0       | 6.8 | 3.4       | 5.0       | 21.9     |
| TS-LSD    | 0.2       | 0.6       | 1.1 | 0.5       | 3.4       | 22.7     |
| TS-HAWP   | 22.1      | 26.6      | 31.1| 22.1      | 26.6      | 31.1     |
| e2vid-HAWP| 22.2      | 26.4      | 31.1| 22.2      | 26.4      | 31.1     |
| L-CNN‡    | 40.6      | 45.8      | 51.0| 40.6      | 45.8      | 51.0     |
| HAWP‡     | 45.1      | 50.4      | 55.6| 45.1      | 50.4      | 55.6     |
| ULSD‡     | 47.0      | 52.7      | 58.2| 47.0      | 52.7      | 58.2     |
| LETR‡     | 24.7      | 34.7      | 44.6| 24.7      | 34.7      | 44.6     |
| FE-HAWP   | 48.7      | 53.9      | 60.4| 48.7      | 53.9      | 60.4     |
| FE-ULSD   | 50.9      | 56.5      | 62.2| 50.9      | 56.5      | 62.2     |

† deblur using the official RED-Net [54] and retrain the line detection model on the de-blurred images.
‡ deblur using the retrained RED-Net [54] and detect line segments using official models.
§ retrain with the concatenation of image and EST.
51.7%. These improvements demonstrate that the event data can enhance the edge awareness for motion-blurred images. Finally, we obtain the highest performance for all metrics using our proposed frame-event fusion module. The msAPs of FE-HAWP and FE-USLD are higher than the retrained ULSD‡ for 1.3 and 3.7 points, respectively. Except for AP\textsuperscript{H}, FE-ULSD obtains the highest accuracy, while AP\textsuperscript{H} is slightly lower than FE-HAWP. This is caused by the low recall rate of FE-ULSD when the confidence threshold is set to 0.5, as shown in the AP\textsuperscript{H} PR curves.

Regarding the detection speed, both FE-ULSD and FE-HAWP are slower than the original methods because their feature fusion backbone has one more encoder branch than the original stacked hourglass network and introduces more time-consuming Transformers for feature fusion. Nevertheless, compared with the transformer-based method LETR [8], the proposed method is about three times faster, which demonstrate the effectiveness of the proposed method.

2) Results on FE-Blurframe Dataset: To further demonstrate the effectiveness of the proposed method in the real-world, we show the qualitative results on FE-Blurframe dataset in Fig. 17. The conventional LSD and FBSD have consistent performances...
as on the synthetic FE-Wireframe dataset. The officially trained models, HAWP and LETR, still have large missing detections but with higher accuracies than the same model on the synthetic FE-Wireframe dataset. This is because of the significant data differences between synthetic and real-world datasets. Then retraining on the concatenation of images and events greatly improves the performance and yields more detected line segments. Similar to the results on FE-Wireframe, the proposed FE-HAWP and FE-ULSD detect the most line segments with the highest similarity to the ground truth labels, reflecting the effectiveness of the proposed method.

Table VIII and Fig. 18 give the quantitative comparisons on the FE-Blurframe dataset. FE-HAWP and FE-ULSD still achieve the best accuracy compared with other learning-based methods. FE-HAWP obtains 52.0% msAP, 9.4 points higher than the retrained HAWP†. FE-ULSD obtains 51.8% msAP, 5.1 points higher than the retrained ULSD†. Because the Bezier equipartition representation of line segments in ULSD is more sensitive to image noises than the attraction field representation in HAWP, the gain of FE-ULSD is not as significant as
FE-HAWP for real images. Furthermore, if we use the transfer learning strategy of fine-tuning on the pre-trained models, i.e., FE-HAWP$^p$ and FE-ULSD$^p$, the performances are improved by 11.3 and 11.1 points compared with FE-HAWP and FE-ULSD, respectively. Additionally, since fewer line candidates are obtained after the fine-tuning, the inference FPS of FE-HAWP$^p$ and FE-ULSD$^p$ are higher than FE-HAWP and FE-ULSD, respectively.

### E. Robustness Analyses

**Robustness to Motion Blur:** By fusing image and event data, we effectively improve the performance of line segment detection on motion-blurred images. However, it is still hard to determine the performance of the fusion over diverse camera motions. Therefore, we analyze the motion blur degree by counting the average displacement of junctions. The distributions of displacement and the corresponding line detection performance are plotted in Fig. 19, where FE-HAWP and HAWP are evaluated. The $x$-axis of each subfigure measures the blurriness by calculating the average displacement of junctions from images. If the mean displacement is close to 0 pixels, it means that the blurriness is small. The blurriness distributions of FE-Wireframe and FE-Blurframe are shown at the top of Fig. 19. The mean blurriness is about 28 pixels for FE-Wireframe and 18 pixels for FE-Blurframe, respectively. The msAP distributions of official HAWP indicate that larger blurs will result in lower line detection accuracy. Then when using the proposed FE-HAWP on images and events, the line detection accuracy is much higher and more robust to the different blurriness.

**Robustness to High Dynamic Range (HDR):** The HDR property of event cameras should also contribute to robust line segment detection. Therefore, we further generate the synthetic HDR images using Wireframe and extend experiments on the synthetic Wireframe dataset. For both dark and bright clear images, FE-HAWP has more true positive detections compared with HAWP. Quantitative comparisons are given in the top part of Table IX, the accuracy of FE-HAWP is much higher than that of HAWP for both dark-clear and light-clear images. Additionally, to comprehensively test the two characteristics of the event camera, i.e., low latency and HDR, we further simulate images with both motion blur and exposure issues, the results are shown in Fig. 20 and the bottom of Table IX. It can be observed that motion blur decreases HAWP accuracy dramatically and most line segments are missing. While for FE-HAWP, our design keeps a high detection rate and the qualitative visualization is much better. Another observation is that FE-HAWP performs better in dark images than in bright images, which means the image coding quantization under strong illumination leads to a loss of information. Moreover, although we do not emphasize the HDR issue in real-world data collection, many HDR images are naturally involved in the FE-Blurframe dataset. The results in Table VIII demonstrate its effectiveness over motion blur and HDR issues.

### V. Conclusion

In this paper, we presented a line segment detection method using events to address the performance degradation in motion-blurred images. The proposed frame-event feature fusion backbone fully exploits the complementary information between images and events with shallow and multi-scale decoder fusion modules. Therefore, the structural edge information can

---

**Table VIII**

Quantitative Comparisons on the FE-Blurframe Dataset

| Method     | $eAP^{15}$ | $eAP^{15/1}$ | $mAP^{15}$ | $mAP^{15/1}$ | $AP^{15}$ | FPS  |
|------------|------------|--------------|------------|--------------|----------|------|
| LSD [6]    | 1.1        | 2.8          | 4.1        | 2.7          | 5.1      | 29.4 | 61.0 |
| FBSD [7]   | 0.9        | 1.9          | 2.7        | 1.8          | 5.1      | 34.2 | 15.9 |
| L-CNN [15] | 7.5        | 11.5         | 13.7       | 10.9         | 12.4     | 27.9 | 29.7 |
| HAWP [9]   | 8.4        | 12.8         | 15.3       | 12.2         | 12.4     | 32.0 | 38.1 |
| ULSD [10]  | 6.8        | 10.8         | 13.0       | 10.2         | 11.8     | 26.7 | 40.6 |
| LETR [8]   | 7.1        | 13.0         | 16.8       | 12.3         | 12.1     | 30.2 | 3.6  |
| TS-LSD     | 0.0        | 0.1          | 0.2        | 0.3          | 2.9      | 43.6 | 62.4 |
| TS-HAWP    | 27.8       | 35.3         | 38.4       | 33.9         | 32.0     | 63.3 | 36.2 |
| c2vid-HAWP | 28.0       | 35.1         | 38.5       | 34.0         | 32.2     | 63.5 | 36.6 |

† pre-train on FE-Wireframe and fine-tune on FE-Blurframe.

---

**Table IX**

Quantitative Results of the HDR Property on Line Segment Detection

| Method                  | $eAP^{5}$ | $eAP^{15}$ | $mAP^{15}$ | $mAP^{15/1}$ | FPS |
|-------------------------|-----------|------------|------------|--------------|-----|
| HAWP (dark-clear)       | 45.2      | 49.1       | 51.0       | 48.4         | 46.0 | 36.6 |
| FE-HAWP (dark-clear)    | 57.1      | 61.4       | 63.4       | 60.7         | 56.7 | 21.1 |
| HAWP (bright-clear)     | 38.6      | 42.2       | 44.0       | 41.6         | 39.2 | 36.6 |
| FE-HAWP (bright-clear)  | 47.0      | 51.3       | 53.5       | 50.6         | 48.7 | 21.4 |
| HAWP (dark-blur)        | 7.5       | 12.1       | 15.2       | 11.6         | 11.0 | 36.6 |
| FE-HAWP (dark-blur)     | 47.2      | 52.5       | 54.9       | 51.5         | 48.1 | 21.4 |
| HAWP (bright-blur)      | 7.7       | 5.0        | 8.0        | 10.0         | 6.7  | 36.6 |
| FE-HAWP (bright-blur)   | 33.9      | 38.8       | 41.0       | 37.9         | 37.4 | 21.3 |

Authorized licensed use limited to the terms of the applicable license agreement with IEEE. Restrictions apply.
be well extracted regardless of slow or fast camera motions. The state-of-the-art line segment detectors, HAWP and ULSD, are then employed to conduct the end-to-end line segment detection. To train our model and contribute to the community, two line segment detection datasets, i.e., FE-Wireframe and FE-Blurframe, are constructed with motion-blurred images and spatial-temporally aligned event data. Extensive analyses of the component configurations validate the reasonability of our network design, and comprehensive comparisons to the state-of-the-arts demonstrate the effectiveness of the proposed FE-LSD for detecting line segments in motion-blurred images with events. Moreover, the proposed method shows robustness when detecting line segments in high dynamic range scenarios. In the future, we will further explore the asynchronous characteristics of event data to simultaneously detect line segments and estimate line motions.

REFERENCES

[1] M. Jin, G. Meishvili, and P. Favaro, “Learning to extract a video sequence from a single motion-blurred image,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., 2018, pp. 6334–6342.
[2] S. Liu, Y. Yu, R. Pautrat, M. Pollefeys, and V. Larsson, “3D line mapping revisited,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., 2023, pp. 21445–21455.
[3] R. Gomez-Ojeda, F.-A. Moreno, D. Zuniga-Noël, D. Scaramuzza, and J. Gonzalez-Jimenez, “PL-SLAM: A stereo SLAM system through the combination of points and line segments,” IEEE Trans. Robot., vol. 35, no. 3, pp. 734–746, Jun. 2019.
[4] Q. Wang, T. Han, Z. Qin, J. Gao, and X. Li, “Multitask attention network for lane detection and fitting,” IEEE Trans. Neural Netw. Learn. Syst., vol. 33, no. 3, pp. 1066–1078, Mar. 2022.
[5] F. Munir, S. Azam, M. Jeon, B.-G. Lee, and W. Pedrycz, “LDNet: End-to-end lane marking detection approach using a dynamic vision sensor,” IEEE Trans. Intell. Transp. Syst., vol. 23, no. 7, pp. 9318–9334, Jul. 2022.
[6] R. Grompone von Gioi, J. Jakubowicz, J.-M. Morel, and G. Randall, “LSD: A fast line segment detector with a false detection control,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 32, no. 4, pp. 722–732, Apr. 2010.
[7] P. Even, P. Ngo, and B. Kerautret, “Thick line segment detection with fast directional tracking,” in Proc. Int. Conf. Image Anal. Process., 2019, pp. 159–170.
[8] Y. Xu, W. Xu, D. Cheung, and Z. Tu, “Line segment detection using transformers without edges,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., 2020, pp. 4255–4264.
[9] N. Xue et al., “Holistically-attracted wireframe parsing,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., 2020, pp. 2788–2797.
[10] H. Li, H. Yu, J. Wang, W. Yang, L. Yu, and S. Scherer, “ULSD: Unified line segment detection across pinhole, fisheye, and spherical cameras,” ISPRS J. Photogrammetry Remote Sens., vol. 178, pp. 187–202, 2021.
[11] W. Xu and S. McCloskey, “2D barcode localization and motion deblurring using a flutter shutter camera,” in Proc. IEEE Workshop Appl. Comput. Vis., 2011, pp. 159–165.
[12] K. Purhot, A. Shah, and A. Rajagopalan, “Bringing alive blurred moments,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., 2019, pp. 6830–6839.
[13] J. Debled-Rennesson, F. Feschet, and J. Rouyer-Degli, “Optimal blurred segments decomposition of noisy shapes in linear time,” Comput. Graph., vol. 30, no. 1, pp. 30–36, 2006.
[14] L. Bazer, “A simple algorithm for digital line recognition in the general case,” Pattern Recognit., vol. 40, no. 6, pp. 1675–1684, 2007.
[15] Y. Zhou, H. Qi, and Y. Ma, “End-to-end wireframe parsing,” in Proc. IEEE/CVF Int. Conf. Comput. Vis., 2019, pp. 962–971.
[16] G. Gallego et al., “Event-based vision: A survey,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 44, no. 1, pp. 154–180, Jan. 2022.
[17] S. Tulyakov et al., “Time lens: Event-based video frame interpolation,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., 2021, pp. 16150–16159.
[18] P. Duan, Z. W. Wang, X. Zhou, Y. Ma, and B. Shi, “EventZoom: Learning to denoise and super resolve neuromorphic events,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., 2021, pp. 12819–12828.
[19] S. Seifozzakerini, W.-Y. Yau, B. Zhao, and K. Mao, “Event-based hough transform in a spiking neural network for multiple line detection and tracking using a dynamic vision sensor,” in Proc. Brit. Mach. Vis. Conf., 2016, pp. 94.1–94.12.
[20] C. Brandli, J. Strubel, S. Keller, D. Scaramuzza, and T. Delbrück, “ELiSed — an event-based line segment detector,” in Proc. 2nd Int. Conf. Event-Based Control, Commun. Signal Process., 2016, pp. 1–7.
[21] Z. Xu, B.-S. Shin, and R. Klette, “Accurate and robust line segment extraction using minimum entropy with hough transform,” IEEE Trans. Image Process., vol. 24, no. 3, pp. 813–822, Mar. 2015.
[22] E. J. Almazán, R. Tal, Y. Qiao, and J. H. Elder, “MCMLSD: A dynamic programming approach to line segment detection,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., 2017, pp. 5854–5862.
[23] C. Akinlar and C. Topal, “EDLines: A real-time line segment detector with a false detection control,” Pattern Recognit. Lett., vol. 32, no. 13, pp. 1633–1642, 2011.
[24] N.-G. Cho, A. Yuille, and S.-W. Lee, “A novel linelet-based representation for line segment detection,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 40, no. 5, pp. 1195–1208, May 2018.
[25] K. Huang, Y. Wang, Z. Zhou, T. Ding, S. Gao, and Y. Ma, “Learning to parse wireframes in images of man-made environments,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., 2018, pp. 626–635.
[26] N. Xue et al., “Learning regional attraction for line segment detection,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 43, no. 6, pp. 1998–2013, Jun. 2021.
[27] Z. Zhang et al., “PPGNet: Learning point-pair graph for line segment detection,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., 2019, pp. 7098–7107.
Q. Meng, J. Zhang, Q. Hu, X. He, and J. Yu, “LGNN: A context-aware line segment detector,” in Proc. 28th ACM Int. Conf. Multimedia, 2020, pp. 4364–4372.

Y. Lin, S. L. Pintea, and J. C. van Gemert, “Deep hough-transform line detection,” in Proc. Eur. Conf. Comput. Vis., 2020, pp. 323–340.

P. Pautrat, J.-T. Lin, V. Larsson, M. R. Oswald, and M. Pollefeys, “SOLD2: Self-supervised occlusion-aware line description and detection,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., 2021, pp. 11363–11373.

I. Debled-Rennesson, R. Jean-Luc, and J. Rouyer-Degli, “Segmentation of discrete curves into fuzzy segments,” Electron. Notes Discrete Math., vol. 12, pp. 372–383, 2003.

B. Kerautret and P. Even, “Blurred segments in gray level images for interactive line extraction,” in Proc. Int. Workshop Combinatorial Image Anal., 2009, pp. 176–186.

D. Reverter Valeiras, X. Clady, S.-H. Ieng, and R. Benosman, “Event-based line fitting and segment detection using a neuromorphic visual sensor,” IEEE Trans. Neural Netw. Learn. Syst., vol. 30, no. 4, pp. 1218–1230, Apr. 2019.

F. Tschopp et al., “Hough Map–iterative event-based hough transform for high-speed railway mapping,” IEEE Robot. Automat. Lett., vol. 6, no. 2, pp. 2745–2752, Feb. 2021.

L. Everding and J. Conradt, “Low-latency line tracking using event-based dynamic vision sensors,” Front. Neurorobot., vol. 12, pp. 1–13, 2018.

W. Cheng, H. Luo, W. Yang, L. Yu, S. Chen, and W. Li, “DET: A high-resolution DVS dataset for lane extraction,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. Workshops, 2019, pp. 1666–1675.

A. Dietsche, G. Ciofﬁ, J. Hidalgo-Carrió, and D. Scaramuzza, “Powerline tracking with event cameras,” in Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst., 2021, pp. 6990–6997.

A. I. Maqueda, A. Loquercio, G. Gallego, N. García, and D. Scaramuzza, “Event-based visual scene understanding on driving cars,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., 2018, pp. 989–997.

D. Gehrig, A. Loquercio, K. Derpanis, and D. Scaramuzza, “End-to-end learning of representations for asynchronous event-based data,” in Proc. IEEE/CVF Int. Conf. Comput. Vis., 2019, pp. 5632–5642.

K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., 2016, pp. 770–778.

S. Woo, J. Park, J.-Y. Lee, and I. S. Kweon, “CBAM: Convolutional block attention module,” in Proc. Eur. Conf. Comput. Vis., 2018, pp. 3–19.

J. Hu, L. Shen, and G. Sun, “Squeeze-and-excitation networks,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., 2018, pp. 7132–7141.

A. Newell, K. Yang, and J. Deng, “Stacked hourglass networks for human pose estimation,” in Proc. Eur. Conf. Comput. Vis., 2016, pp. 483–499.

J. Guo et al., “CMT: Convolutional neural networks meet visual transformers,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., 2022, pp. 12175–12185.

A. Vaswani et al., “Attention is all you need,” in Proc. Adv. Neural Inf. Process. Syst., 2017, pp. 1–11.

H. Rebecca, D. Gehrig, and D. Scaramuzza, “ESIM: An open event camera simulator,” in Proc. 2nd Conf. Robot. Learn., 2018, pp. 969–982.

Z. W. Wang, P. Duan, O. Cossairt, A. Katsaggelos, T. Huang, and B. Shi, “Joint ﬁltering of intensity images and neuromorphic events for high-resolution noise-robust imaging,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., 2020, pp. 1606–1616.

H. Jiang, D. Sun, V. Jampali, M.-H. Yang, E. Learned-Miller, and J. Kautz, “Super SlowMo: High-quality estimation of multiple intermediate frames for video interpolation,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., 2018, pp. 9000–9008.

D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” in Proc. Int. Conf. Learn. Representations, 2015, pp. 1–15.

H. Rebecca, R. Ranftl, V. Koltun, and D. Scaramuzza, “High speed and high dynamic range video with an event camera,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 43, no. 6, pp. 1964–1980, Jun. 2021.

F. Xu et al., “Motion deblurring with real events,” in Proc. IEEE/CVF Int. Conf. Comput. Vis., 2021, pp. 2583–2592.