EventTracer: Fast Path Tracing-Based Event Stream Rendering

Zhenyang Li, Xiaoyang Bai, Jinfan Lu, Pengfei Shen, Edmund Y. Lam, *Fellow, IEEE*, and Yifan Peng

***Abstract*—Simulating event streams from 3D scenes has become a common practice in event-based vision research, as it meets the demand for large-scale, high temporal frequency data without setting up expensive hardware devices or undertaking extensive data collections. Yet existing methods in this direction typically work with noiseless RGB frames that are costly to render, and therefore their simulations are often unrealistically low in temporal resolution. In this work, we propose *EventTracer*, a path tracing-based rendering pipeline that simulates high-fidelity event sequences from complex 3D scenes in an efficient and physics-aware manner. Specifically, we speed up the rendering process via low sample-per-pixel (SPP) path tracing, and train a lightweight event spiking network to denoise the resulting RGB videos into realistic event sequences. Our EventTracerpipeline runs at a speed of ∼1 minutes per second of 360p video, and it inherits the merit of accurate spatiotemporal modeling from its path tracing backbone. We show through the *Real2Sim* and *Sim2Real* tests that EventTracercaptures higher-fidelity scene details and demonstrates a greater similarity to real-world event data than alternative event simulators, which establishes it as a potential tool for creating large-scale event-RGB datasets, narrowing the sim-to-real gap in event-based vision, and boosting various downstream applications.**

***Index Terms*—Event camera, event simulation, path tracing, spiking neural network.**

I. Introduction

OVER the past few years, event streams have emerged as a unique modality that compensates the conventional RGB videos with high temporal resolution and motion sensitivity [1]. In the fields of drone navigation, autonomous driving, robotics, and AR/VR, event sensory have been widely integrated into the perception and decision pipeline to enhance their performance in dynamic and challenging environments [2], [3], [4]. Nowadays, event-based vision is dominated by learning-based methods [5], which effectively tackle the noisy nature of event streams. Yet, training those models poses a high demand for large-scale event-RGB datasets, whose availability is still far from sufficient.

Compared to unimodal RGB videos, event-RGB data are noticeably more costly and time-consuming to collect. Commercial event cameras are usually expensive, and they either lack frame-based imaging capability (e.g. Prophesee's EVK series and iniVation's DVXplorer) or feature low resolution and inferior imaging fidelity [6]. While a few datasets have been curated using calibrated and synchronized multi-camera systems [7], [8], they often fail to address the demand for task-specific annotations, such as depth maps, surface normals, and segmentation masks. Including those modalities will necessitate extra hardware setups or algorithmic post-processing, which further complicates the data collection pipeline.

To address this issue, prior studies [9], [10] have explored synthetic event generation from RGB videos, requiring clear intensity frames as input. Such a constraint limits their temporal resolution to equivalently 100∼300 FPS, even coupled with frame interpolation algorithms, considerably lower than the one of actual event cameras (≥1,000 FPS). While a higher frame rate is attainable by running simulations on videos rendered from 3D scenes [11], [12], [13], such an approach is yet limited by the tradeoff between speed and quality. On one end sits rasterization, which renders at an impeccable speed but lacks photorealism, especially under complex lighting conditions; conversely, the time consumption of noise-free global illumination techniques such as path tracing can pile up to several seconds per frame, making the entire process intolerably slow. Efficient, realistic, and high temporal resolution event simulation is yet an unsolved problem.

In light of this, we propose *EventTracer*, a path tracing-based rendering pipeline that satisfies all three requirements. To speed up rendering, we apply Monte Carlo path tracing with a low sample-per-pixel (SPP) setting. Next, we train a *lightweight event spiking network (EvSNet)* to emulate event signals from noisy pixel illuminances. To capture the physical properties of event sensors, we propose a bipolar leaky integrate-and-fire (BiLIF) unit to activate the outputs of EvSNetinto event pulses, and train the whole network with a bidirectional earth mover's distance (EMD) loss. Unlike other learning-based approaches [10], the spatial and temporal localness of EvSNetallows us to integrate it into Falcor [14] using NVIDIA TensorRT [15] and fully parallelize the rendering process. The overall EventTracerpipeline is able to simulate event streams from dynamic 3D scenes at 1,000 FPS with a speed of 1.12 minutes per second of 360p video (Fig. 1 *left*), surpassing prior

Received 28 August 2025; revised 28 May 2026; accepted 2 June 2026. Date of publication 8 June 2026; date of current version 3 July 2026. This work was supported in part by the National Science Foundation of China under Grant 62322217, in part by the Research Grants Council of Hong Kong under Grant GRF 17208023, and in part by the Innovation and Technology Fund of Hong Kong under Grant ITP/062/24AP. Recommended for acceptance by Y. Dong. *(Zhenyang Li and Xiaoyang Bai contributed equally to this work.) (Corresponding author: Yifan Peng.)*

Zhenyang Li, Xiaoyang Bai, Pengfei Shen, Edmund Y. Lam, and Yifan Peng are with The University of Hong Kong, Hong Kong (e-mail: evanpeng@hku.hk).

Jinfan Lu is with The University of Hong Kong, Hong Kong, and also with the Tsinghua University, Beijing 100084, China.

This article has supplementary downloadable material available at https://doi.org/10.1109/TVCG.2026.3701141, provided by the authors.

Digital Object Identifier 10.1109/TVCG.2026.3701141

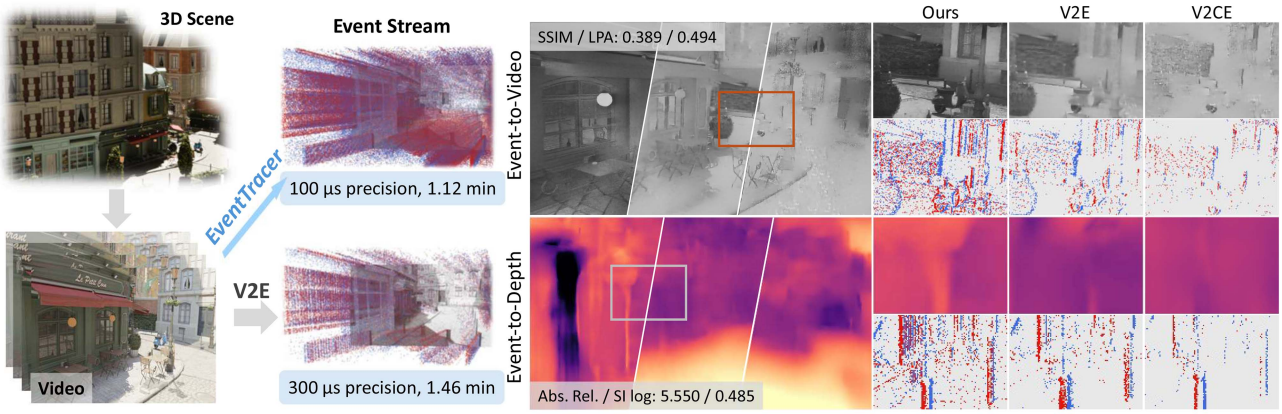


Fig. 1. EventTracerpipeline overview. *Left:* Event generation from 3D scenes with existing event simulators such as V2E is bottlenecked by the trade-off between rendering speed and temporal resolution, as they operate exclusively on noiseless RGB frames; our proposed EventTracerbreaches such limitations and achieves 3× temporal precision with only 76% rendering time. *Right:* EventTraceroutperforms V2E and V2CE on the *Real2Sim* test for two downstream tasks, event-to-video and event-to-depth, as it preserves more details in complex 3D scenes.

works in both runtime and temporal resolution. Besides, its outputs are high dynamic range and easily extended by extra annotations such as segmentation masks and depth maps, thanks to the path tracing mechanism.

To evaluate the effectiveness of EventTracer, we use it to build an event-RGB dataset, named ETScenes, that consists of 90 seconds of videos rendered from 6 different 3D scenes. Using this dataset, we compare EventTracerwith V2E [9], V2CE [10] and DVS-Voltmeter [16] in two complementary tests, *Real2Sim* and *Sim2Real*, on both event-to-video reconstruction [17] and event-based depth estimation [18]. Experiment results demonstrate that our simulated ETScenesis more similar to real-captured data than its peers, validating the superior render quality of EventTracer, in addition to its efficiency (Fig. 1 *right*). Therefore, EventTracercan serve as a powerful tool for efficiently generating high-quality synthetic event-RGB data in the near future, meeting the needs of various event-based vision tasks and narrowing the sim-to-real gap that bottlenecks the research community. In summary, our main technical contributions include:

- We propose a path tracing-based event rendering framework, dubbed *EventTracer*, that realizes fast and accurate simulation of event-RGB data from 3D scenes.
- We design *EvSNet*, a lightweight physics-aware neural network that can be seamlessly integrated with the path tracer to realize fully-differentiable event rendering.
- We propose an evaluation protocol consisting of the *Real2Sim* and the *Sim2Real* tests to quantitatively compare among different event simulation methods.
- We construct the *ETScenes* dataset with EventTracerand apply the evaluation protocol on different downstream vision tasks to demonstrate its improved quality and similarity to real-world data, compared to those produced by existing evenet simulators.

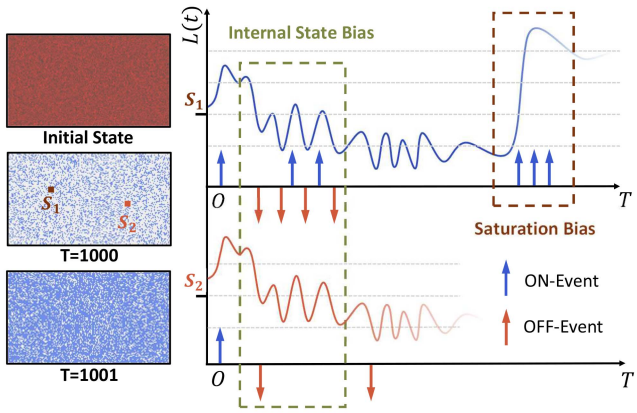


Fig. 2. *Left:* (top) Each pixel is initialized with a randomly sampled internal state; (bottom) Identical brightness changes are applied to all pixels, causing them to fire events at different timestamps. *Right:* **Internal State Bias**: Two event pixels with different initial states (S_1 and S_2) produce drastically different event pulses (**blue** and **red** arrows) even when they undergo identical illuminance changes; **Saturation Bias**: instant changes in illuminance create trailing events afterwards.

II. Related Work

A. Event Data Collection and Synthesis

Event cameras are novel sensory systems that asynchronously measure per-pixel illuminance variations [19], well-suited for challenging scenarios involving high-speed motion or power constraints. Recently, their application has extended beyond the conventional high-speed imaging [20], [21] into 3D vision [22], intelligent camera control [23], [24], and event-based optics [25], which increasingly levitates its prospect in fields such as robotics, autonomous driving, and wearable electronics [1]. With this trend, numerous datasets with paired RGB-event data have been released, including CED [26], LED [27], and EventAid [8]. Some datasets also include additional annotations, for example, depth maps in MVSEC [28] and segmentation masks in EV-IMO [29]. Alternative approaches like CIFAR10-DVS [30] image RGB videos played on high refresh rate screens with event cameras, which saves the effort of multi-camera calibration and synchronization. Yet all these datasets require costly hardware setups and time-consuming collection processes, and they are often limited in resolution, scene diversity, and video length, potentially hindering their effectiveness for downstream tasks.

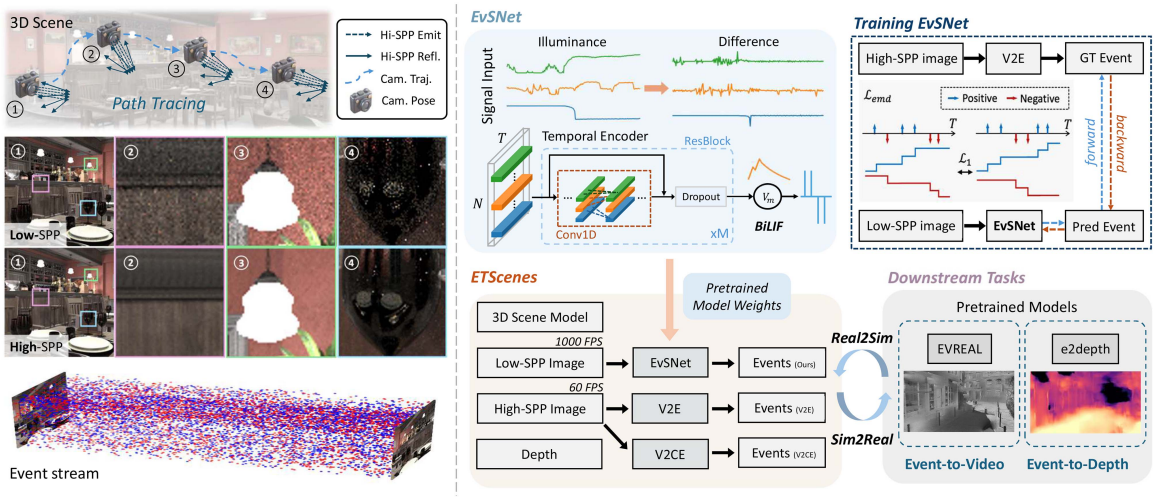


Fig. 3. Pipeline of our *EventTracer*. *Bottom left:* Our task is to directly render event streams from 3D scenes. *Top-left:* In each scene, we define a camera trajectory and run the path tracer at varying sample-per-pixel (SPP) to obtain low-SPP (noisy, high FPS) and high-SPP (noiseless, low FPS) RGB videos. *Top-right:* Our EvSNetnetwork takes noisy pixel illuminances as input and outputs discrete event spikes via the BiLIF unit; it is trained with V2E-simulated events, using an earth mover's distance (EMD) loss. *Bottom-right:* While existing event simulators (e.g., V2E, V2CE, DVS-Voltmeter) can only run on high-SPP videos, EvSNetis able to convert low-SPP inputs to event streams. We validate its functionality through the *Real2Sim* and *Sim2Real* tests, conducted on two downstream tasks.

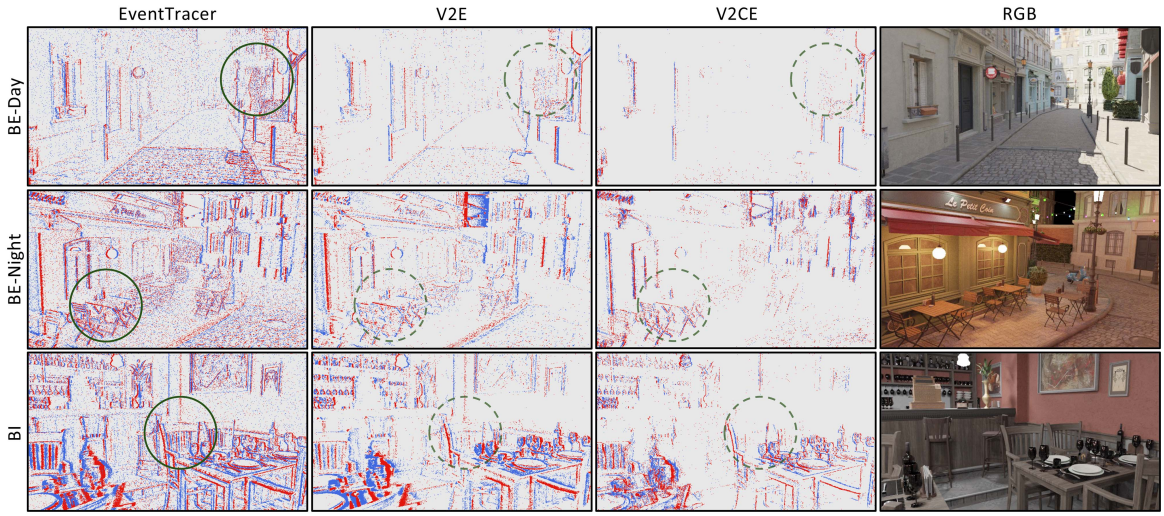


Fig. 4. Comparison of events obtained using various rendering approaches–EventTracer, V2E, and V2CE–on three scenes: Bistro-Exterior-Day (BE-Day), Bistro-Exterior-Night (BE-Night) and Bistro-Interior (BI). We also provide the corresponding RGB frames for reference.

Given this, there has been a growing interest in synthetic event generation, which can be categorized into two types based on: *existing video sequences* and *3D simulated scenes*. In the former category, [9] introduced V2E, a toolbox that synthesizes realistic events from gray-scale videos which models various aspects of actual event sensors, while DVS-Voltmeter [16] leverages a stochastic process to inject circuit properties into simulated event streams. More recently, [10] developed V2CE as a learning-based tool to generate continuous event streams from videos. However, the temporal resolution of these video-based methods is limited by the relatively low frame rate of input videos (typically $\leq$60 FPS). Even with the help from pretrained frame interpolation models [31], the final temporal sensitivity still struggles to meet downstream application needs.

Conversely, simulation-based methods leverage physically accurate 3D models to generate synthetic event data under fully controlled conditions. Notable works include ESIM [11], BlinkFlow [13], and PECS [12]. Although these simulators deliver high temporal resolution results that are easily extended by extra annotations, their efficiency is constrained by the backbone rendering speed. For example, noiseless (2,048 SPP) path tracing runs at 1.36 seconds per 360p frame, suggesting >20 minutes of runtime for rendering 1,000 FPS videos and subsequently simulating high-fidelity events. Thus, rendering temporally dense, high-resolution event streams that match the μs precision of event cameras in real time remains challenging.

Closely related to our approach are two recent papers [32], [33] which propose to adaptively denoise or sample from

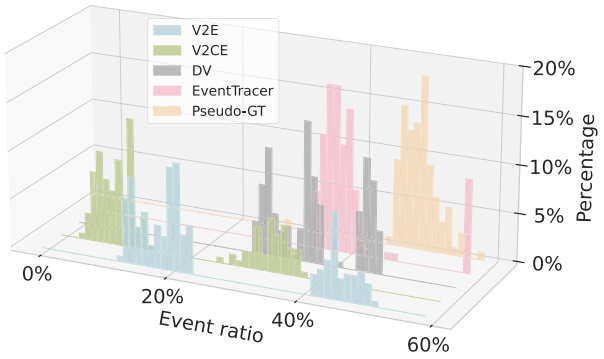


Fig. 5. Histograms of event ratio (percentage of pixels with event activations in each frame) for four investigated simulators and the pseudo-ground truth. We take 60 frames with 30 FPS from each of the three Amazon Lumberyard Bistro scenes.

low-SPP Monte Carlo path tracing to simulate realistic event streams. With these pioneering works proving the feasibility of speeding up event rendering with reduced SPP, we further advance this concept by leveraging a learnable denoising network to simulate sensor-end noise factors such as *internal state bias* and *saturation effect*, as well as to relax the requirements for smooth local luminance change [32] or sparse event triggering [33], boosting the noise modeling capacity and performance stability of path tracing-based event simulation to a new level.

B. Spiking Neural Network (SNN) for Event-Based Vision

The concept of SNN has been explored for over two decades [34], [35]. As a bio-inspired architecture that mimics neuron activities in human brains, SNNs are well-suited for processing spatiotemporal data [36] at a reduced computation cost than ordinary artificial neural networks (ANNs) [37], [38]. Since spiking units such as variants of leaky integrate-and-fire (LIF) [39] are non-differentiable, various strategies to back propagate through SNNs have been developed, including the prevalent ANN-to-SNN conversion [40], [41] and the surrogate gradient method [42], [43]. As a result, SNN is becoming increasing popular in computer vision and deep learning in general [44], [45].

The neuromorphic design and low-cost properties of SNNs naturally draw researchers towards applying them on event streams. A hybrid ANN-SNN architecture is usually employed to process event data into desired outputs [46], [47], and SNN-empowered methods have demonstrated success across multiple downstream tasks, including video reconstruction [48], classification [49], action recognition [50], and optical flow estimation [51]. To the best of our knowledge, EventTracerpioneers in leveraging spiking units to *generate*, instead of process, event streams.

C. Real2Sim and Sim2Real Evaluation of Synthetic Data

It is a common practice to evaluate synthetic data by directly or indirectly comparing them against their real-world counterparts, especially in the absence of reliable quantitative metrics. The *Sim2Real* test operates by training models with simulated data and evaluating them on real-world scenarios, and it has been widely used in the field of robotics [52] and more recently visual-language models [53]. The complementary *Real2Sim* test (with a slight abuse of terminology) trains models with real-world data and investigates their performance on synthetic inputs. In this paper, we validate our EventTracerpipeline and compare it against other event simulators following an evaluation protocol combining the *Real2Sim* and *Sim2Real* tests, inspired by their applications in assessing large language models [54] and robot simulation systems [55].

III. Method

A. Motivation: Noise Factors in Event Sensory

Unlike conventional frame-based sensors that record absolute pixel illuminance via exposure, event sensors constantly monitor each pixel's illuminance and fire a spike (i.e., an event) whenever its logarithmic change exceeds a predefined contrast threshold ρ. As a result, an event camera outputs a sequence of asynchronous spikes, each denoted by a tuple (t, x, y, p), where t denotes the timestamp, (x, y) are the pixel coordinates, and $p \in \{+1, -1\}$ indicates the polarity of the change (positive or negative). This design grants event sensors advantages such as extremely high temporal resolution ($<$100 μs) and high dynamic range, but it also introduces unique noise compositions to the resulting event streams. Realistically reproducing the distribution of real-world events has thus become a key challenge of event synthesis.

a) Internal state bias: The asynchronous nature of event sensor causes its pixel circuits to have different *voltage levels (i.e., internal states)* at each moment. In other words, two event pixels may produce drastically different events during a given time interval, even when their illuminance changes are identical, as illustrated in Fig. 2. Such an internal state bias is the main factor why real-world event streams appear noisy to human eyes.

As a result, inferring event occurrences solely from the illuminance difference between two consecutive frames, a common practice undertaken by existing event simulators [11], almost certainly results in suboptimal performance. To resolve this issue, our proposed EvSNetnetwork scans through the entire temporal span of the input clip and keeps the internal state recorded within its BiLIF activation unit. This approach realistically simulates the behavior of event cameras and ensures a consistent distribution of events across the entire video.

b) Saturation effect: After firing an event spike, the event pixel undergoes a reset period before it can give further responses. Therefore, when the instant illumination change is very large, it will take the sensor a relatively long time to fire all events, which creates a sequence of pulses in the observed event stream, as visualized in Fig. 2. We call this phenomenon the *saturation effect*. Failing to address such an effect in simulation will increase the domain gap between synthetic and real-world events, especially around extremely bright or extremely dark regions, and further impair the utility of generated event data on downstream tasks. While this effect has been addressed by some physics-based simulation methods [9], reproducing it with artificial neural network (ANN) modules would require a large

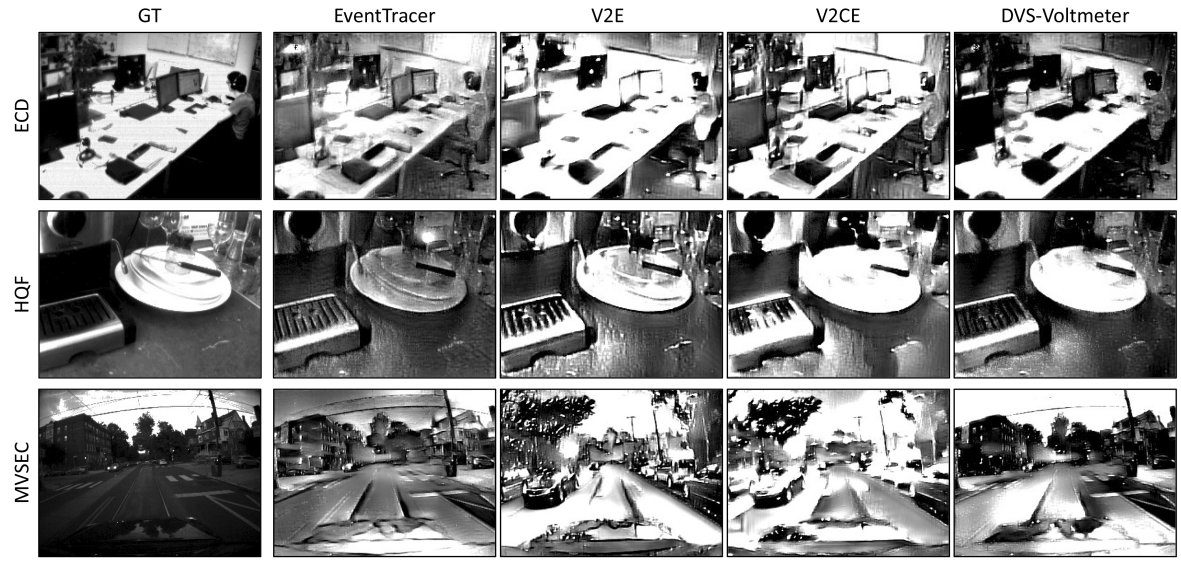


Fig. 6. Visual comparison of *Sim2Real* results. Note that the E2V model trained with ETScenes(simulated by EventTracer) reconstructs fine scene details such as plate edges (HQF) and overhead wires (MVSEC).

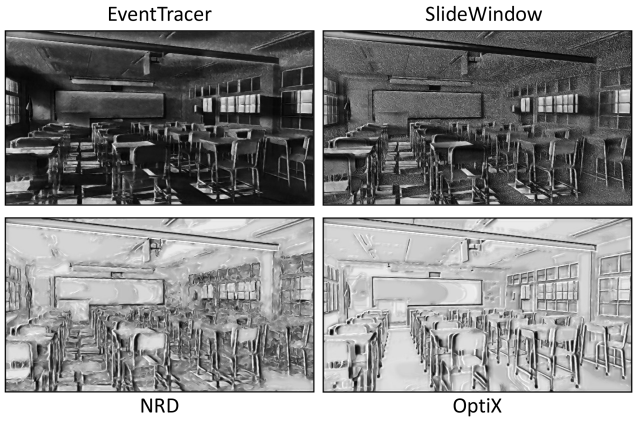


Fig. 7. Visualization of reconstructed frames with events simulated by our EventTracer, SlideWindow, NRD and OptiX. FireNet+ is used as the E2V model.

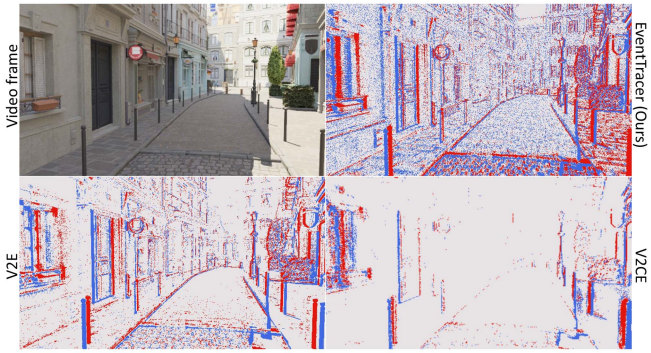


Fig. 8. Event simulation under fast motion. Observe that EventTracergenerates trailing event activations for the leftmost window bars, which form thick stripes when integrated. This is a clear evidence of saturation effect being simulated.

receptive field and longer training time, and thus it is often ignored by existing learning-based approaches [10].

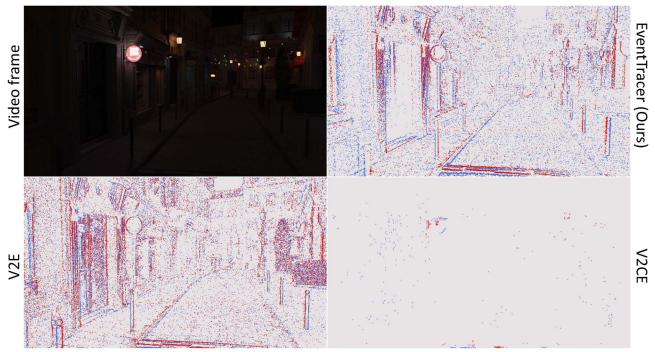


Fig. 9. Event simulation under extreme darkness. Like real-world event cameras, EvSNettakes HDR pixelwise log-difference as input, and therefore EventTraceris agnostic to absolute scene brightness.

To this end, we develop the *bipolar leaky integrate-and-fire (BiLIF) spiking unit* that naturally exhibits the saturation effect without explicit training. Specifically, spiking neurons produce output activations *one at a time*, and therefore a large incoming signal caused by instant drastic luminance change would automatically be broken down to a sequence of spikes that trail beyond the current timestamp, precisely reproducing the behavior of event pixels. On the other hand, learning such a behavior with convolution networks would require a large receptive field and extensive training steps. More details on BiLIF are available in Section III-C.

c) Other noise factors: Beyond threshold bias and saturation, event streams also experience minor perturbations from event camera circuitry, such as threshold mismatch, hot pixels, leak noise events, and shot noise [9]. In our pipeline, low-SPP Monte Carlo path tracing naturally introduces shot-noise–like variations in the RGB input, and our EvSNetnetwork also helps to emulate these effects by learning from the outputs of the physics-based V2E method, resulting in synthesized events that closely match the noise characteristics of real sensors.

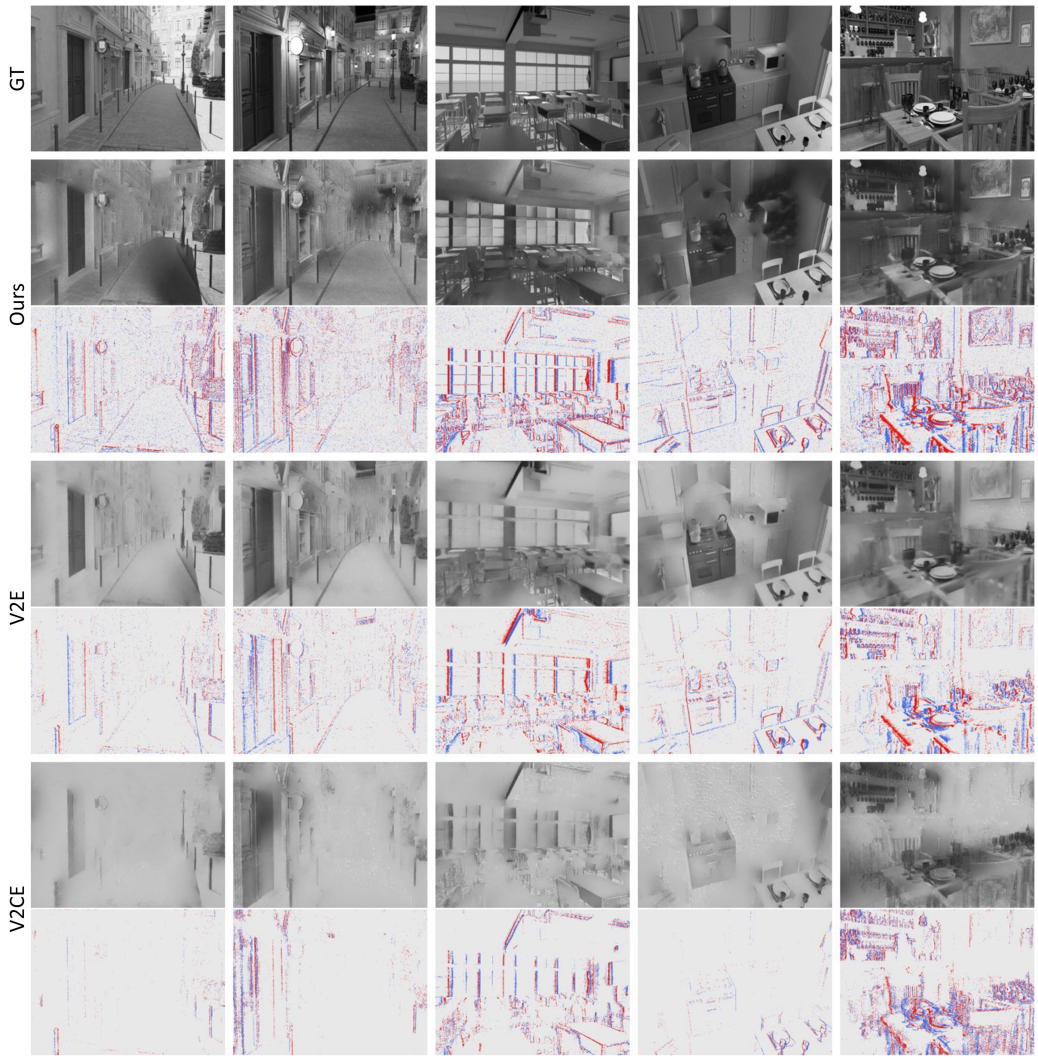


Fig. 10. *Real2Sim* validation results on E2V. Each column contains the ground truth grayscale frame as well as pairs of reconstructed image and event frame for each event simulator. When evaluated on our ETScenesdataset, the pretrained FireNet [71] model is able to reconstruct more details in the scene, including lamp posts, window grids, chairs, and paintings. In contrast, evaluation results on ETScenes-V2E and ETScenes-V2CE contain large blank regions, suggesting information loss in their event data. While denser event patterns may preserve additional scene details, they do not alone imply higher physical realism; these visualizations are complemented by statistical and sim-to-real evaluations, which is same to Fig. 11.

B. EventTracer: Path Tracing-Based Event Rendering

To synthesize event streams from 3D scenes, we firstly discretize time over a duration of T seconds with a high frame rate F_{evt} into $K = T \cdot F_{\text{evt}}$ timestamps. Then for each $t_k = k/F_{\text{evt}}$, we render an RGB image $\mathbf{I}_k$ with *Monte Carlo path tracing*. While various other global illumination (GI) methods—for example, Lumen in UE5, DLSS and NRD [56]—may be used for real-time, noise-free rendering, their temporal accumulation strategies tend to blur or suppress edge features in scenes with fast-moving objects, making them suboptimal choices for simulating event streams that are particularly sensitive to edge movements.

To make sure simulated event data has sufficiently high temporal resolution, we choose $F_{\text{evt}} = 1,000$. However, conventional high sample-per-pixel (SPP) path tracing (at least $N = 2,048$ SPP is needed to produce noiseless RGB frames) is impractical under this setting due to its excessive time consumption. Therefore, we decrease the SPP to $N = 64$ in our pipeline, reducing the runtime of the path tracer to $1/32$. The resulting sequence $\{\mathbf{I}_0, \mathbf{I}_1, \ldots, \mathbf{I}_K\}$ is then a high-FPS yet noisy approximation of real-world capture. As simply computing events by frame difference cannot account for the various noise factors in real-world event streams, we instead train a *pixel-wise event pulsing network (EvSNet)* to simultaneously denoise path tracing results and predict event spikes. Specifically, for each pixel location (x, y), we extract the sequence of pixel values $\{I_0^{(x,y)}, \ldots, I_K^{(x,y)}\}$ and convert it to logarithmic differences of illuminance as follows:

$$L = 0.2126\, I_R + 0.7152\, I_G + 0.0722\, I_B\,,$$

$$L^{\log} = \begin{cases} \ln L\,, & L \geq \rho \\ \frac{\ln \rho}{\rho} L\,, & L < \rho \end{cases}, \quad X_k = L_k^{\log} - L_{k-1}^{\log}. \tag{1}$$

Here $L \in [0, 255]$, ρ is a threshold that is set to 20, and we omit pixel locations (x, y) for clarity. We then input the processed sequence $\mathbf{X} = \{X_1, \ldots, X_K\}$ into EvSNetand obtain simulated events $\mathbf{E} = \{e_1, \ldots, e_K\}$. Details on EvSNetare explained in Section III-C.

Since EvSNetperforms pixel-wise and temporally local (with receptive field width $W < 50$) inference, we are able to integrate it into the Falcor [14] renderer using the NVIDIA TensorRT framework [15] and easily parallelize the computation across the whole frame, achieving a rendering speed of ∼1 minutes per second of 360p events video. The overall pipeline of EventTraceris visualized in Fig. 3.

C. EvSNet: Lightweight Event Spiking Network

a) Spiking neuron preliminaries: Spiking neurons are special activation units that convert sequential inputs into discrete spikes when the membrane potential crosses a certain threshold. A commonly used spiking neuron is the leaky integrate-and-fire (LIF) model [39], whose dynamics are described as:

$$\begin{aligned} (\textbf{charge})\ V'(t) &= (1 - 1/\tau) \cdot V(t-1) + I(t), \\ (\textbf{fire})\ S(t) &= 1\left[V'(t) \geq V_{\text{th}}\right], \\ (\textbf{reset})\ V(t) &= V'(t) - S(t), \end{aligned} \tag{2}$$

where $V(t)$ is the membrane potential, $I(t)$ is the synaptic input current, $S(t)$ is the output spike, and τ is the decay constant. In a word, an output spike is generated when $V(t)$ exceeds the threshold V_{th}, after which $V(t)$ is reset. To enable gradient-based training, a surrogate derivative can be used for the non-differentiable spike function [43].

b) Bipolar LIF (BiLIF) for event simulation: To accommodate the inherently bipolar nature of event camera outputs, we extend the conventional LIF neuron to a *bipolar LIF*, dubbed BiLIF, that is able to emit both positive and negative event spikes. In BiLIF, the charging and resetting of membrane potential $V(t)$ follows the formulation in (2), while the firing step is now governed by two symmetric thresholds, $+V_{\text{th}}$ and $-V_{\text{th}}$. Specifically, the bipolar spike $S(t) \in \{-1, 0, +1\}$ is computed according to:

$$S(t) = \operatorname{sgn}\left[V(t)\right] 1\left(|V(t)| \geq V_{\text{th}}\right), \tag{3}$$

where $\operatorname{sgn}(u) = +1$ if $u > 0$ and -1 if $u < 0$.

This mechanism allows the neuron to simultaneously represent increases and decreases in illuminance as positive and negative pulses, respectively. Similar to the vanilla LIF, we also adopt a surrogate gradient approach for the non-differentiable Heaviside function. Let $\sigma(u)$ be a smooth surrogate function, during backpropagation we can approximate that:

$$\frac{\partial S(t)}{\partial V(t)} \approx \frac{\partial \sigma\left(V(t) - V_{\text{th}}\right)}{\partial V(t)} - \frac{\partial \sigma\left(-V(t) - V_{\text{th}}\right)}{\partial V(t)}. \tag{4}$$

Our training uses the fast arctangent surrogate [57]. This formulation ensures both ON and OFF events contribute meaningful gradients, enabling the network to learn representations that capture the full polarity spectrum of event streams.

c) Network architecture: We design our EvSNetnetwork to be lightweight, pixel-wise, and temporally local so that it can be smoothly integrated into the rendering pipeline. As previously mentioned, it takes as input a sequence $\mathbf{X}$ of logarithmic differences of illuminance signals. Then it passes $\mathbf{X}$ through an 1D convolutional encoder to extract per-timestamp features:

$$\begin{aligned} \mathbf{h}^{(0)} &= \operatorname{ReLU}\left[\operatorname{Conv1d}(\mathbf{X})\right], \\ \mathbf{h}^{(i)} &= \operatorname{ReLU}\left[\mathbf{h}^{(i-1)} + \operatorname{Conv1d}\left(\operatorname{ReLU}\left[\operatorname{Conv1d}(\mathbf{h}^{(\mathbf{i-1})})\right]\right)\right]. \end{aligned} \tag{5}$$

A point-wise convolution is applied to project the feature tensor to a length-K sequence that is subsequently used to excite event pulses via a BiLIF spiking unit:

$$\mathbf{S} = \operatorname{BiLIF}\left[\operatorname{Conv1d}\left(\mathbf{h}^{(M)}\right)\right] \in \{-1, 0, 1\}^{B \times K}. \tag{6}$$

The temporal receptive field of EvSNetis relatively narrow, and thus the renderer only needs to refer to a small set of neighboring frames to infer the event distribution at the current timestamp. This design effectively reduces the memory and time consumption of the EventTracerpipeline.

d) Loss functions and training:: We render high-quality, F_{evt} FPS videos with high-SPP path tracing and feed them into the V2E toolbox [9] to generate ground truth events $\mathbf{E}$. To align the predicted event sequence $\mathbf{S}$ with $\mathbf{E}$, both of which are sparse and discrete-valued, we construct the loss as the 1D earth mover's distance (EMD) between them. It is straightforward to deduce that the EMD between two equal-length 1D sequences is the pairwise distance between their sorted entries [58]. Therefore, the loss can be computed as:

$$\mathcal{L}(\mathbf{E}, \mathbf{S}) = \frac{1}{K} \sum_{i=1}^{K} \left\| \sum_{j=1}^{i} \mathbf{S}_j - \sum_{j=1}^{i} \mathbf{E}_j \right\|_1, \tag{7}$$

that is, the difference between the cumulative sums of $\mathbf{S}$ and $\mathbf{E}$ at each timestamp in the sequence. However, this formulation presumes that $\mathbf{E}$ and $\mathbf{S}$ have the same number of nonzero entries. While this condition is not inherently met by the EvSNetnetwork, Eq. 7 will prompt the model to hallucinate nonexistent events towards the end of the sequence. To address this drawback, we compute the EMD twice in different directions as:

$$\mathcal{L}'(\mathbf{E}, \mathbf{S}) = \frac{1}{2}\left[\mathcal{L}(\mathbf{E}, \mathbf{S}) + \mathcal{L}(\mathbf{E}^{\text{rev}}, \mathbf{S}^{\text{rev}})\right], \tag{8}$$

where $\mathbf{E}^{\text{rev}}$ and $\mathbf{S}^{\text{rev}}$ are the reversed sequences.

Considering that $\mathbf{E}$ and $\mathbf{S}$ contain both positive and negative values, we separate them into two channels and apply (8) individually to each channel:

$$\mathcal{L}_{\text{EMD}}(\mathbf{E}, \mathbf{S}) = \mathcal{L}'\left(\mathbf{E}_{>0}, \mathbf{S}_{>0} \cdot \mathbf{S}\right) + \mathcal{L}'\left(\mathbf{E}_{<0}, \mathbf{S}_{<0} \cdot (-\mathbf{S})\right). \tag{9}$$

To further constrain EvSNeton the total number of event spikes, we additionally regularize training with an event spike count loss:

$$\mathcal{L}_{\text{count}} = \left\| \sum_{i=1}^{K} |\mathbf{S}_i| - \sum_{i=1}^{K} |\mathbf{E}_i| \right\|_1. \tag{10}$$

The total loss is the sum of two terms with a weight λ:

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{EMD}} + \lambda \cdot \mathcal{L}_{\text{count}}. \tag{11}$$

The overall EventTracerframework is easily realizable with popular renderers such as Falcor or UE5. Furthermore, it can be straightforwardly combined with *denoising algorithms* such as NRD and OptiX [59] as well as *runtime speedup tricks* such as ignoring indirect light or removing environment maps. In this way, its efficiency under less challenging scenarios can be further boosted.

IV. Assessment

A. Implementation Details

EvSNet is trained with paired low-SPP RGB videos and high-fidelity event streams. Since ground truth events are unavailable for rendered scenes, we instead construct *pseudo-ground truths* by running V2E on high-SPP, high-FPS inputs. Specifically, we render 2∼4 seconds of 2,048 SPP, 1,000 FPS video for each of the three scenes in Amazon Lumberyard Bistro [60], *i.e.*, Bistro-Interior, Bistro-Exterior-Day and Bistro-Exterior-Night, amounting to 7,936 frames in total. While constructing the training set alone takes >12 hours, the trained EvSNetnetwork and the overall EventTracerpipeline can be readily applied to unseen data, reducing its amortized time consumption for each new scene rendered.

The constructed pseudo-ground truth establishes a theoretical upper bound of rendering-based event simulation, which our EvSNetis trained to approach with far less SPP. In other words, the goal of EventTraceris exactly to strike a balance between speed and quality, as will be proven experimentally below.

We set the kernel size for all but the last convolution layers to 7 and the network depth $M = 3$, resulting in a relatively small temporal receptive field width of $W = 43$. BiLIF is implemented using the SpikingJelly framework [61]. The model is trained for 20 epochs and all experiments are conducted on a NVIDIA GeForce RTX 4090 GPU. The resolution of rendered images is $1,280 \times 720$, while for downstream tasks we downsample it to 640×360, since most existing baselines and pretrained models are tuned for low-resolution inputs. Additional implementation details can be found in the Supplementary Material.

a) The ETScenesdataset: After training the EvSNetnetwork, we construct the *ETScenes* dataset with paired event-RGB videos to validate our EventTracerpipeline. Given a dynamic 3D scene, we render event stream with EventTracerand its corresponding RGB video with regular high SPP path tracing at a frame rate of 60 FPS. A total of 6 scenes are used: three from Amazon Lumberyard Bistro and three from [62], namely Kitchen, Staircase and Classroom. We render 20 seconds of videos for Amazon Lumberyard Bistro scenes, and 10 seconds for the remaining scenes, resulting in an overall 90 seconds of videos with ∼6,000 RGB frames and (equivalently) ∼90,000 event "frames". Additionally, we also render ground truth depth maps corresponding to each RGB frame with almost no extra computation cost. We note that other annotations, such as surface normals, optical flows and segmentation masks, can be easily obtained likewise.

B. Validation Setup

Quantitatively measuring an event stream's fidelity can be challenging, as raw events are not readily interpretable or visually understandable to humans [17]. To this end, our full evaluation protocol assesses the fidelity of ETScenesand compares EventTraceragainst other event simulators from three aspects: event ratio distribution (Section IV-C), event stream statistics (Section IV-D), and a combination of the *Real2Sim* and *Sim2Real* tests conducted on two downstream tasks, *event-to-video reconstruction (E2V)* (Section IV-E) and *event-to-depth estimation (E2D)* (Section IV-F). Specifically, the *Real2Sim* test evaluates baseline models pretrained with existing real-world data $\mathcal{D}_{\text{gt}}$ on different simulated datasets $\mathcal{D}_{\text{sim}}$, and the *Sim2Real* test trains new models on $\mathcal{D}_{\text{sim}}$ and evaluate them on $\mathcal{D}_{\text{gt}}$. In both settings, better results indicate a greater similarity between $\mathcal{D}_{\text{gt}}$ and $\mathcal{D}_{\text{sim}}$ and serve as a surrogate metric for evaluating the performance of event simulators.

a) Event-to-video reconstruction (E2V): One of the most fundamental tasks of event-based vision, event-to-video aims at reconstructing frame-based intensity images from input event streams. We adopt the EVREAL benchmark [17] for the *Real2Sim* test, which supports the evaluation of 8 baseline E2V models on the given data using multiple quantitative metrics. For the *Sim2Real* experiment, we train E2VID+ [63] models with simulated event data and conduct evaluation on MVSEC [64], ECD [65], and HQF [66].

b) Event-to-depth reconstruction (E2D): Recovering per-pixel depth maps directly from asynchronous event streams has emerged as a key challenge in neuromorphic vision, promising robust range sensing under high-speed and high dynamic range conditions. We perform *Real2Sim* validation using the e2depth model [18] pretrained on mixed DENSE (synthetic) and MVSEC (real) [64] datasets.

c) Comparison to other event simulators: We compare EventTraceragainst three mainstream event simulators, V2E [9], V2CE [10] and DVS-Voltmeter [16], and render the counterparts of ETScenesusing each method, denoted as ETScenes-V2E, ETScenes-V2CE and ETScenes-DV, respectively, from RGB videos whose frame rate does not exceed 60 FPS for a fair comparison. Specifically, we observe that V2CE does not work with 30 FPS inputs where adjacent frames are too similar. Therefore, we have to increase frame difference by downsampling the input videos to 15 FPS, so that ETScenes-V2CE contains non-trivial event streams. A qualitative comparison across ETScenes, ETScenes-V2E and ETScenes-V2CE is available in Fig. 4, where we can readily observe that EventTracerpreserves more scene details, especially high-frequency patterns that produce quickly alternating event signals as the camera moves.

C. Event Ratio Distribution

We inspect the distribution of event ratio, *i.e.*, the percentage of pixels with event activations in a frame, of each dataset as a primitive way of evaluating its simulation quality. As shown in Fig. 5, ETScenesfeatures the most concentrated distribution which resembles the pseudo-ground truth, while other baseline

TABLE I
EVENT STREAM STATISTICS. WE REPORT THE MEAN AND STANDARD DEVIATION OF EACH METRIC OVER 6 SCENES AND 12 INTERVALS PER SCENE.

Method	RMSE↓	RMSE Std↓	KL↓	KL Std↓	Corr↑	Corr Std↓
EventTracer	**0.071**	**0.017**	**0.833**	**0.238**	**0.830**	**0.058**
V2E	0.112	0.022	4.657	1.301	0.472	0.099
V2CE	0.117	0.021	7.165	1.694	0.581	0.073
DVS-Voltmeter	0.098	0.023	1.266	0.457	0.711	0.075

datasets exhibit a greater dispersion across selected frames. Moreover, the **Wasserstein distance** to the pseudo-GT histogram for ETScenesis measured to be ∼32.4% lower than that of ETScenes-DV, and ∼64% lower than that of ETScenes-V2E and ETScenes-V2CE, numerically supporting our observation from Fig. 5.

D. Event Stream Statistics

We then directly assess whether simulated events are spatially aligned with the underlying scene illuminance changes. To this end, we compare the illuminance change between consecutive grayscale frames and the corresponding accumulated event frames simulated by different methods. We report three statistical metrics: root mean squared error (RMSE) between normalized frames, KL divergence [67] between their spatial distributions, and pixel-wise Pearson correlation (Corr) [68].

For each scene, we evaluate 12 representative high-activity intervals and compute both the mean and standard deviation of the above statistics. Table I reports average scores over all six scenes. Specifically, standard deviations of each metric quantify temporal stability, where a low value indicates that the simulator remains consistent across different motion phases rather than performing well on only a few intervals.

From the table, we observe that EventTracerconsistently outperforms other baselines in both event-scene alignment and simulation stability. Detailed plots of event count curves and that of illuminance change for all scenes are provided in the Supplementary Material.

E. Event-to-Video Reconstruction

Since event streams only record relative changes in illuminance, the illuminance level of videos reconstructed by E2V models can vary from case to case, largely affecting the scores of various metrics and making them less indicative of the actual image quality. To resolve this issue, we apply histogram equalization (HE) on both ground truth and reconstructed RGB frames to map their overall illuminance to the same level.

For the *Real2Sim* test, we report structural similarity (SSIM), learned perceptual image patch similarity (LPIPS) [69], and energy of image Laplacian (LPA) [70] between reconstructed images and ground truth after HE. We refrain from using pixel-wise intensity metrics such as mean square error (MSE) or peak signal-to-noise ratio (PSNR) due to their high sensitivity to image illuminance level. Results are compared in Table II. It can be observed that events simulated by EventTraceralign better with models pretrained on real-world data, resulting in higher-quality and sharper video reconstruction. A qualitative visualization is provided in Fig. 10, where EventTracerdemonstrates its ability to retain high-frequency features such as street tiles and window grids, while the reconstructed images from ETScenes-V2E and ETScenes-V2CE exhibit large featureless regions due to the loss in visual details.

For the *Sim2Real* test, we report SSIM and LPIPS for reconstructed images after HE. Results are compared in Table. III. Note that EvSNetparallels DVS-Voltmeter in *Sim2Real* performance, and they both surpass v2e and V2CE by a large margin. Since all three target datasets are captured with real-world event cameras, these results clearly demonstrate that the E2V model trained on ETScenesis able to generalize to real-world event data, subsequently proving its superior simulation quality. See Fig. 6 for a visual comparison.

F. Event-to-Depth Estimation

The depth distribution of our ETScenesdataset, featuring mostly indoor scenes, is drastically different from that of autonomous driving datasets such as DENSE and MVSEC. Therefore, we only run the *Real2Sim* test and evaluate relative error metrics to quantify the performance of e2depth models, including absolute relative error (Abs.Rel.), square relative error (Sq.Rel.), linear RMSE (RMSE), logarithmic RMSE (RMSE log), and scale-invariant logarithmic error (SI.log).

Table IV quantitatively compares the performance of different event simulators. Our EventTracerachieves top-tier scores, on par with DVS-Voltmeter and significantly surpassing v2e and V2CE, in all metrics, again demonstrating its capability to produce realistic event streams that can be readily processed by pretrained deep models. A qualitative comparison is available in Fig. 11. We can observe that the pretrained e2depth model identifies more scene layouts, such as desks, stairs and bar counters, and makes more consistent depth estimations for those objects when evaluated on ETScenes.

G. Comparison to Off-the-Shelf Denoisers

We have also compared EventTraceragainst three off-the-shelf denoisers: NRD [56], OptiX [59], and a simple sliding window averager that smoothens the value of each pixel temporally (denoted as "SlideWindow" hereafter). Specifically, we firstly render high-FPS and low-SPP videos with each denoiser active, and then calculate event frames as the log-difference between two consecutive frames. We report their *Real2Sim* performance on the E2V task in Table V, and visualize a sample reconstructed frame for each method in Fig. 7. We attribute the outstanding performance of EventTracer to the fact that off-the-shelf denoisers result in overly smooth RGB frames and do not account for the unique noise patterns in event streams (Section III-A). Yet, as noted in Section III, those algorithms can be easily integrated with EventTracer under suitable circumstances to enable faster event rendering. We leave this to future exploration.

H. Performance in Adverse Conditions

Event cameras are particularly advantageous in challenging scenarios such as fast motion and extreme lighting. Here we

TABLE II
QUANTITATIVE COMPARISON UNDER THE *REAL2SIM* SETTING USING MULTIPLE E2V BASELINES. METRICS (FROM LEFT TO RIGHT): SSIM↑, LPIPS↓ AND LPA ($\times 10^3$) ↑.

Methods	E2VID	E2VID+	FireNet	FireNet+	SPADE–E2VID	SSL–E2VID	ET–Net	HyperE2VID
V2CE	0.244 / 0.648 / 0.23	0.238 / 0.513 / **0.44**	0.215 / 0.607 / 0.19	0.218 / 0.578 / 0.74	0.242 / 0.550 / 0.16	0.193 / 0.705 / 0.14	0.238 / 0.525 / **0.50**	0.240 / 0.518 / 0.66
V2E	**0.351** / 0.501 / 0.14	0.395 / 0.410 / 0.18	0.370 / 0.472 / 0.09	0.326 / 0.457 / 0.58	0.356 / 0.421 / 0.15	0.202 / 0.674 / 0.06	0.398 / 0.400 / 0.25	0.393 / 0.395 / 0.33
DVS-Voltmeter	0.258 / 0.661 / 0.20	0.316 / 0.537 / 0.28	0.351 / 0.555 / 0.30	0.217 / 0.577 / 1.03	0.362 / 0.551 / 0.31	**0.337** / 0.722 / 0.15	0.284 / 0.548 / 0.46	0.271 / 0.549 / 0.50
EventTracer	0.348 / **0.478** / **0.24**	**0.413** / **0.375** / 0.32	**0.424** / **0.400** / **0.42**	**0.384** / **0.399** / **1.16**	**0.405** / **0.404** / **0.40**	0.289 / **0.605** / **0.29**	**0.409** / **0.387** / 0.42	**0.438** / **0.350** / **0.70**

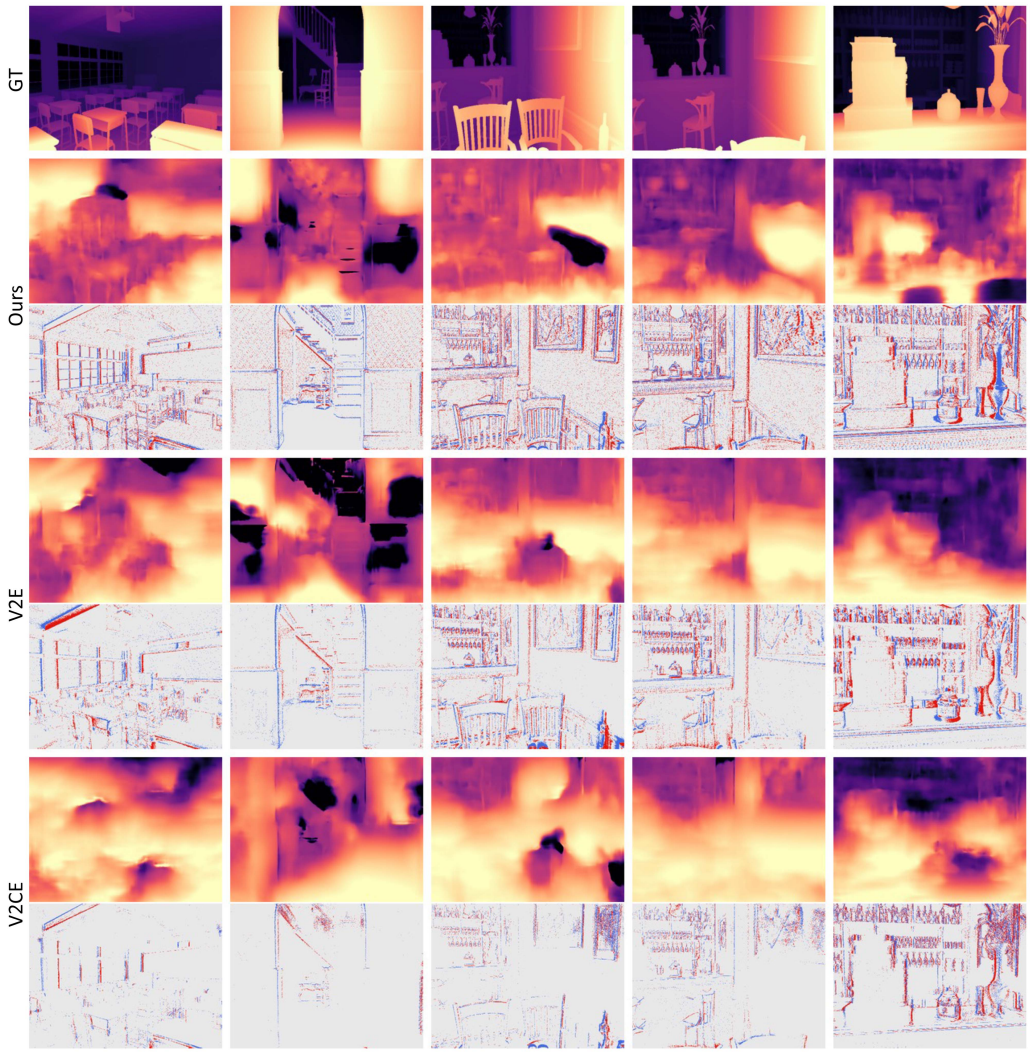


Fig. 11. *Real2Sim* validation results on E2D. Each column contains the ground truth depth map as well as pairs of estimated depth and event frame for each event simulator. Note that ETScenesyields the best results of the three, where not only detailed structures (e.g. desks, stairs, and bar counters) are visible, but the overall depth estimation for each scene also displays certain consistency. When using ETScenes-V2E and ETScenes-V2CE, more details are lost and scene layouts are barely distinguishable from the output depth maps.

TABLE III
QUANTITATIVE COMPARISON UNDER THE *SIM2REAL* SETTING. SCORES FOR THE ORIGINAL E2VID+ MODEL ARE ALSO PROVIDED FOR REFERENCE. METRICS: SSIM↑ (LEFT) AND LPIPS↓ (RIGHT).

Dataset	EventTracer	V2E	V2CE	DVS-Voltmeter	E2VID+
MVSEC	0.216 / 0.613	0.164 / 0.633	0.141 / 0.655	**0.222** / **0.587**	0.223 / 0.522
ECD	0.268 / 0.440	0.237 / 0.479	0.234 / 0.544	**0.339** / **0.431**	0.323 / 0.361
HQF	**0.365** / **0.411**	0.330 / 0.436	0.272 / 0.489	0.353 / 0.436	0.451 / 0.287

stress-test EventTracerunder these adverse conditions to demonstrate its likeness to real-world event cameras.

Fast motion: We speed up the camera trajectory in Bistro-Exterior-Day by 10× to simulate fast motion and visualize output events in Fig. 8. Notably, EventTracerclearly exhibits saturation effect as introduced in Section III-A thanks to its spiking BiLIF layer. In contrast, the output of v2e is indistinguishable from its slow-motion counterpart, while V2CE simply fails to

TABLE IV
QUANTITATIVE COMPARISON UNDER THE *REAL2SIM* SETTING USING PRETRAINED *E2DEPTH* NETWORK.

Method	Abs. Rel. ↓	Sq. Rel. ↓	RMSE ↓	RMSE log ↓	SI. log ↓
V2CE	0.898	6.304	29.172	1.393	0.582
V2E	0.990	9.917	28.225	1.284	0.529
DVS-Voltmeter	**0.880**	**4.775**	28.202	**1.141**	**0.439**
EventTracer	0.896	5.550	**28.170**	1.147	0.485

TABLE V
QUANTITATIVE COMPARISON BETWEEN EVENTTRACERAND OFF-THE-SHELF DENOISERS UNDER THE *REAL2SIM* SETTING FOR THE E2V TASK. METRICS USED ARE SSIM↑ (LEFT) AND LPIPS↓ (RIGHT). SCORES ARE AVERAGED OVER ALL BASELINES IN THE EVREAL BENCHMARK.

Methods	SSIM↑	LPIPS↓
SlideWindow	0.379	0.434
NRD	0.381	0.452
OptiX	0.355	0.464
EventTracer	**0.389**	**0.425**

resolve the domain gap between slow-paced training data and fast-moving input frames.

Low light: We create challenging low-light scenarios by dimming the luminance level of Bistro-Exterior-Night even more. Observe in Fig. 9 that EventTracerand v2e both reproduces the robustness of real-world event cameras in extremely dark environments.

I. Rendering Speed

We compare the rendering speed of each method by reporting the time it takes to render *one second of 360p event video*. With 60 FPS RGB videos as input, V2E takes 1.85 minutes (1.46 min for RGB rendering and 0.39 min for event simulation), V2CE takes 1.53 minutes (1.48 min for RGB rendering and 0.05 min for event simulation), and DVS-Voltmeter takes 1.61 minutes (1.46 min for RGB rendering and 0.15 min for event simulation), while our EventTraceronly takes 1.12 minutes, 39.4% faster than V2E, 26.8% faster than V2CE, and 30.4% faster than DVS-Voltmeter.

V. CONCLUSION

In this work, we have presented a novel path tracing-based event rendering pipeline, dubbed EventTracer, realizing fast and high-fidelity simulation of event data from 3D scenes. This pipeline consists of an efficient low-SPP Monte Carlo path tracing renderer and a lightweight event spiking network, featuring the BiLIF spiking unit and trained with the earth mover's distance loss to capture the physical properties of event signals. We compare EventTraceragainst existing event simulators by constructing synthetic event-RGB datasets and evaluating them on two downstream tasks with the *Real2Sim* and *Sim2Real* tests. Experiment results demonstrate that EventTraceroutperforms existing tools such as V2E, V2CE and DVS-Voltmeter in both event stream fidelity and rendering speed, thereby narrowing the sim-to-real gap that hinders a wide range of event-based vision research.

Limitations and future works: While our EvSNetnetwork can be trained to approximate the pixel-wise event distribution with high fidelity, it is yet only weakly constrained to produce the same number of event pulses as the ground-truth signal. As a result, its outputs may induce degraded performance in downstream tasks that are sensitive to event counts, for example, event-to-video reconstruction. It is left for future works to explore additional loss terms and/or training strategies to mitigate this issue.

Besides, we may also explore the integration between EventTracerand other denoising/sampling methods to achieve even higher simulation fidelity. On the RGB rendering side, NRD and OptiX rank among the most competitive candidates, while on the event generation side, [32] and [33] have proposed inspiring learning-free solutions for noise mitigation in path tracing-based event simulation. In addition, as a fully differentiable and path tracing-based solution to event synthesis, EventTraceralso inspires us to explore the potential of fusing event data with *differentiable optics*, where researchers develop designated point spread functions (PSFs) and subsequently utilize these PSFs to encode target images [72], [73]. Advances in this direction will equip event sensors with better imaging quality and improved sensitivity to depth and motion, ultimately reshaping the field of event-based vision.

While it is possible to augment existing event simulators to model *internal state bias* and/or *saturation effect* (for example, by processing their outputs with the BiLIF layer), such an exploration is well beyond the scope of this paper and we have to leave it to future research.

ACKNOWLEDGMENT

The authors thank Dr. Shijie Lin for fruitful discussions.

REFERENCES

[1] G. Gallegoet al., "Event-based vision: A survey," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 44, no. 1, pp. 154–180, Jan. 2022.
[2] A. N. Angelopoulos, J. N. P. Martel, A. P. S. Kohli, J. Conradt, and G. Wetzstein, "Event based, near-eye gaze tracking beyond 10,000 Hz," *IEEE Trans. Visual. Comput. Graph.*, vol. 27, no. 5, pp. 2577–2586, May 2021.
[3] J. Xuet al., "Taming event cameras with bio-inspired architecture and algorithm: A case for drone obstacle avoidance," in *Proc. 29th Annu. Int. Conf. Mobile Comput. Netw.*, 2023, pp. 1–16.
[4] D. Gehrig and D. Scaramuzza, "Low-latency automotive vision with event cameras," *Nature*, vol. 629, no. 8014, pp. 1034–1040, 2024.
[5] X. Zhenget al., "Deep learning for event-based vision: A comprehensive survey and benchmarks," 2023, *arXiv:2302.08890*.
[6] C. Brandli, R. Berner, M. Yang, S.-C. Liu, and T. Delbruck, "A 240 × 180 130 db 3 μs latency global shutter spatiotemporal vision sensor," *IEEE J. Solid-State Circuits*, vol. 49, no. 10, pp. 2333–2341, Oct. 1914.
[7] X. Wanget al., "VisEvent: Reliable object tracking via collaboration of frame and event flows," *IEEE Trans. Cybern.*, vol. 54, no. 3, pp. 1997–2010, Mar. 1924.
[8] P. Duanet al., "EventAid: Benchmarking event-aided image/video enhancement algorithms with real-captured hybrid dataset," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 47, no. 8, pp. 6959–6973, Aug. 2025.
[9] Y. Hu, S.-C. Liu, and T. Delbruck, "V2E: From video frames to realistic DVS events," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. Workshops*, Jun. 2021, pp. 1312–1321.
[10] Z. Zhang et al., "V2CE: Video to continuous events simulator," in *Proc. IEEE Int. Conf. Robot. Automat.*, May 2024, pp. 12455–12461.
[11] H. Rebecq, D. Gehrig, and D. Scaramuzza, "ESIM: An open event camera simulator," in *Proc. 2nd Conf. Robot Learn.*, Oct. 2018, pp. 969–982.

[12] H. Han et al., "Physical-based event camera simulator," in *Proc. Eur. Conf. Comput. Vis.*, 2024, pp. 19–35.

[13] Y. Li et al., "BlinkFlow: A dataset to push the limits of event-based optical flow estimation," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Oct. 2023, pp. 3881–3888.

[14] S. Kallweit et al., "The falcor rendering framework," 2022. [Online]. Available: https://github.com/NVIDIAGameWorks/Falcor, https://github.com/NVIDIAGameWorks/Falcor

[15] NVIDIA, "Nvidia tensorrt," 2025. [Online]. Available: https://developer.nvidia.com/tensorrt/

[16] S. Lin, Y. Ma, Z. Guo, and B. Wen, "DVS-voltmeter: Stochastic process-based event simulator for dynamic vision sensors," in *Proc. Eur. Conf. Comput. Vis.*, 2022, pp. 578–593.

[17] B. Ercan, O. Eker, A. Erdem, and E. Erdem, "EVREAL: Towards a comprehensive benchmark and analysis suite for event-based video reconstruction," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2023, pp. 3943–3952.

[18] J. Hidalgo-Carrió, D. Gehrig, and D. Scaramuzza, "Learning monocular dense depth from events," in *Proc. Int. Conf. 3D Vis.*, 2020, pp. 534–542.

[19] P. Lichtsteiner, C. Posch, and T. Delbruck, "A 128 × 128 120 db 15 μs latency asynchronous temporal contrast vision sensor," *IEEE J. Solid-State Circuits*, vol. 43, no. 2, pp. 566–576, Feb. 2008.

[20] S.-H. Ieng, J. Carneiro, and R. B. Benosman, "Event-based 3D motion flow estimation using 4D spatio temporal subspaces properties," *Front. Neurosci.*, vol. 10, 2017, Art. no. 596.

[21] Y. Zou, Y. Zheng, T. Takatani, and Y. Fu, "Learning to reconstruct high speed and high dynamic range videos from events," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2021, pp. 2024–2033.

[22] B. Yu, J. Ren, J. Han, F. Wang, J. Liang, and B. Shi, "EventPS: Real-time photometric stereo using an event camera," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2024, pp. 9602–9611.

[23] H. Lou, M. Teng, Y. Yang, and B. Shi, "All-in-focus imaging from event focal stack," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Vancouver, BC, Canada, Jun. 2023, pp. 17366–17375.

[24] S. Lin et al., "Embodied neuromorphic synergy for lighting-robust machine vision to see in extreme bright," *Nature Commun.*, vol. 15, no. 1, 2024, Art no. 10781.

[25] S. Shah et al., "CodedEvents: Optimal point-spread-function engineering for 3D-tracking with event cameras," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2024, pp. 25265–25275.

[26] C. Scheerlinck, H. Rebecq, T. Stoffregen, N. Barnes, R. Mahony, and D. Scaramuzza, "CED: Color event camera dataset," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. Workshops*, Jun. 2019, pp. 1684–1693.

[27] Y. Duan, "LED: A large-scale real-world paired dataset for event camera denoising," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Seattle, WA, USA, Jun. 2024, pp. 25637–25647.

[28] A. Z. Zhu, D. Thakur, T. Ozaslan, B. Pfrommer, V. Kumar, and K. Daniilidis, "The multi vehicle stereo event camera dataset: An event camera dataset for 3D perception," *IEEE Robot. Autom. Lett.*, vol. 3, no. 3, pp. 2032–2039, Jul. 2018.

[29] A. Mitrokhin, C. Ye, C. Fermüller, Y. Aloimonos, and T. Delbruck, "EV-IMO: Motion segmentation dataset and learning pipeline for event cameras," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2019, pp. 6105–6112.

[30] H. Li, H. Liu, X. Ji, G. Li, and L. Shi, "CIFAR10-DVS: An event-stream dataset for object classification," *Front. Neurosci.*, vol. 11, May 2017, Art. no. 309.

[31] H. Jiang, D. Sun, V. Jampani, M.-H. Yang, E. Learned-Miller, and J. Kautz, "Super SloMo: High quality estimation of multiple intermediate frames for video interpolation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 9000–9008.

[32] Y. Tsuji, T. Yatagawa, H. Kubo, and S. Morishima, "Event-based camera simulation using monte carlo path tracing with adaptive denoising," in *Proc. IEEE Int. Conf. Image Process.*, 2023, pp. 301–305.

[33] Y. Manabe, T. Yatagawa, S. Morishima, and H. Kubo, "Monte Carlo path tracing and statistical event detection for event camera simulation," in *Proc. IEEE Int. Conf. Comput. Photogr.*, 2024, pp. 1–10.

[34] W. Gerstner, "Time structure of the activity in neural network models," *Phys. Rev. E*, vol. 51, no. 1, pp. 738–758, 1995.

[35] W. Maass, "Networks of spiking neurons: The third generation of neural network models," *Neural Netw.*, vol. 10, no. 9, pp. 1659–1671, 1997.

[36] A. Tavanaei, M. Ghodrati, S. R. Kheradpisheh, T. Masquelier, and A. Maida, "Deep learning in spiking neural networks," *Neural Netw.*, vol. 111, pp. 47–63, 2019.

[37] P. A. Merolla et al., "A million spiking-neuron integrated circuit with a scalable communication network and interface," *Science*, vol. 345, no. 6197, pp. 668–673, 2014.

[38] J. Pei et al., "Towards artificial general intelligence with hybrid tianjic chip architecture," *Nature*, vol. 572, no. 7767, pp. 106–111, 2019.

[39] W. Gerstner, W. M. Kistler, R. Naud, and L. Paninski, *Neuronal Dynamics: From Single Neurons to Networks and Models of Cognition*. Cambridge, U.K.:Cambridge Univ. Press, 2014.

[40] E. Hunsberger and C. Eliasmith, "Spiking deep networks with LIF neurons," 2015, *arXiv:1510.08829*.

[41] S. Deng and S. Gu, "Optimal conversion of conventional artificial neural networks to spiking neural networks," in *Proc. Int. Conf. Learn. Representations*, 2021.

[42] Y. Wu, L. Deng, G. Li, J. Zhu, and L. Shi, "Spatio-temporal backpropagation for training high-performance spiking neural networks," *Front. Neurosci.*, vol. 12, 2018, Art. no. 331.

[43] E. O. Neftci, H. Mostafa, and F. Zenke, "Surrogate gradient learning in spiking neural networks: Bringing the power of gradient-based optimization to spiking neural networks," *IEEE Signal Process. Mag.*, vol. 36, no. 6, pp. 51–63, Nov. 2019.

[44] C. D. Schuman, S. R. Kulkarni, M. Parsa, J. P. Mitchell, P. Date, and B. Kay, "Opportunities for neuromorphic computing algorithms and applications," *Nature Comput. Sci.*, vol. 2, no. 1, pp. 10–19, 2022.

[45] N. Rathi et al., "Exploring neuromorphic computing based on spiking neural networks: Algorithms to hardware," *ACM Comput. Surv.*, vol. 55, no. 12, pp. 1–49, 2023.

[46] A. Kugele, T. Pfeil, M. Pfeiffer, and E. Chicca, "Hybrid SNN-ANN: Energy-efficient classification and object detection for event-based vision," in *Proc. DAGM German Conf. Pattern Recognit.*, 2021, pp. 297–312.

[47] S. H. Ahmed, J. Finkbeiner, and E. Neftci, "Efficient event-based object detection: A hybrid neural network with spatial and temporal attention," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2025, pp. 13970–13979.

[48] L. Zhu, X. Wang, Y. Chang, J. Li, T. Huang, and Y. Tian, "Event-based video reconstruction via potential-assisted spiking neural network," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2022, pp. 3594–3604.

[49] M. Yao et al., "Temporal-wise attention spiking neural networks for event streams classification," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2021, pp. 10221–10230.

[50] Q. Liu, D. Xing, H. Tang, D. Ma, and G. Pan, "Event-based action recognition using motion information and spiking neural networks," in *Proc. Int. Joint Conf. Artif. Intell.*, 2021, pp. 1743–1749.

[51] J. Hagenaars, F. Paredes-Vallés, and G. De Croon, "Self-supervised learning of event-based optical flow with spiking neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2021, vol. 34, pp. 7167–7179.

[52] S. Höfer et al., "Sim2real in robotics and automation: Applications and challenges," *IEEE Trans. Autom. Sci. Eng.*, vol. 18, no. 2, pp. 398–400, Apr. 2021.

[53] K. Singh, T. Navaratnam, J. Holmer, S. Schaub-Meyer, and S. Roth, "Is synthetic data all we need? Benchmarking the robustness of models trained with synthetic images," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2024, pp. 2505–2515.

[54] V. Veselovsky, M. H. Ribeiro, A. Arora, M. Josifoski, A. Anderson, and R. West, "Generating faithful synthetic data with large language models: A case study in computational social science," 2023, *arXiv:2305.15041*.

[55] X. Li et al., "Evaluating real-world robot manipulation policies in simulation," in *Proc. Conf. Robot Learn.*, 2025, pp. 3705–3728.

[56] NVIDIA, "Nvidia NRD," 2025. [Online]. Available: https://github.com/NVIDIA-RTX/NRD

[57] I. Aliyev and T. Adegbija, "Fine-tuning surrogate gradient learning for optimal hardware performance in spiking neural networks," in *Proc. Des., Automat. Test Europe Conf. Exhib.*, 2024, pp. 1–2.

[58] C. Villani, *Optimal Transport: Old and New*, vol. 338. Berlin, Germany: Springer, 2008.

[59] NVIDIA, "Nvidia optix," 2025. [Online]. Available: https://developer.nvidia.com/rtx/ray-tracing/optix

[60] A. Lumberyard, "Amazon lumberyard bistro, open research content archive (ORCA)," Jul. 2017. [Online]. Available: http://developer.nvidia.com/orca/amazon-lumberyard-bistro

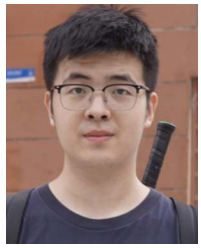
[61] W. Fang et al., "SpikingJelly: An open-source machine learning infrastructure platform for spike-based intelligence," *Sci. Adv.*, vol. 9, no. 40, 2023, Art. no. eadi1480.[Online]. Available: https://www.science.org/doi/abs/10.1126/sciadv.adi1480

[62] B. Bitterli, "Rendering resources," 2016. [Online]. Available: https://benedikt-bitterli.me/resources/
[63] H. Rebecq, R. Ranftl, V. Koltun, and D. Scaramuzza, "Events-to-video: Bringing modern computer vision to event cameras," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2019, pp. 3852–3861.
[64] A. Z. Zhu, D. Thakur, T. Özaslan, B. Pfrommer, V. Kumar, and K. Daniilidis, "The multivehicle stereo event camera dataset: An event camera dataset for 3D perception," *IEEE Robot. Autom. Lett.*, vol. 3, no. 3, pp. 2032–2039, Jul. 2018.
[65] E. Mueggler, H. Rebecq, G. Gallego, T. Delbruck, and D. Scaramuzza, "The event-camera dataset and simulator: Event-based data for pose estimation, visual odometry, and SLAM," *Int. J. Robot. Res.*, vol. 36, no. 2, pp. 142–149, 2017.
[66] T. Stoffregen et al., "Reducing the sim-to-real gap for event cameras," in *Proc. Eur. Conf. Comput. Vis.*, 2020, pp. 534–549.
[67] S. Kullback and R. A. Leibler, "On information and sufficiency," *Ann. Math. Statist.*, vol. 22, no. 1, pp. 79–86, 1951.
[68] K. Pearson, "Note on regression and inheritance in the case of two parents," *Proc. Roy. Soc. London*, vol. 58, pp. 240–242, 1895.
[69] R. Zhang, P. Isola, A. A. Efros, E. Shechtman, and O. Wang, "The unreasonable effectiveness of deep features as a perceptual metric," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 586–595.
[70] M. Subbarao, T.-S. Choi, and A. Nikzad, "Focusing techniques," *Opt. Eng.*, vol. 32, no. 11, pp. 2824–2836, 1993.
[71] C. Scheerlinck, H. Rebecq, D. Gehrig, N. Barnes, R. Mahony, and D. Scaramuzza, "Fast image reconstruction with an event camera," in *Proc. IEEE/CVF Winter Conf. Appl. Comput. Vis.*, 2020, pp. 156–163.
[72] X. Yang, Q. Fu, and W. Heidrich, "Curriculum learning for ab initio deep learned refractive optics," *Nature Commun.*, vol. 15, no. 1, Aug. 2024, Art. no. 6572.
[73] Z. Shi et al., "Learned multi-aperture color-coded optics for snapshot hyperspectral imaging," *ACM Trans. Graph.*, vol. 43, no. 6, pp. 1–11, 2024.

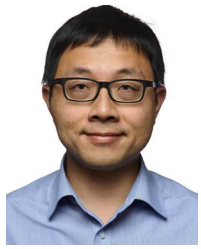
Zhenyang Li received the bachelor's degree in electronic and information engineering from the School of Electronic & Optical Engineering, Nanjing University of Science and Technology, and the master's degree from the Department of Automation, Tsinghua University, specializing in Big Data engineering. He is currently working toward the PhD degree with the Department of Electrical & Electronic Engineering, University of Hong Kong. His research interests include computer vision, computer graphics, and VR/AR/MR. (https://lagrangeli.github.io/)

Xiaoyang Bai received the BA degree in computer science, applied mathematics, cognitive science & linguistics from the University of California, Berkeley, and the PhD degree in computer science from the University of Illinois Urbana-Champaign. He is currently a postdoctoral fellow with the Department of Electrical & Electronic Engineering, The University of Hong Kong. His research interests include event-based vision, computer graphics, and computer vision. (https://andrewbxy.github.io/)

Jinfan Lu is currently working toward the undergraduation degree with the Institute for Interdisciplinary Information Sciences (Yao Class), Tsinghua University. His research interests include real-time & realistic rendering, sampling strategy, and denoising within computer graphics. He aims to advance rendering algorithms and systems to make real-time path tracing truly practical and widely usable.



Pengfei Shen received the master's degree from Geometry Computing Lab, University of Science and Technology of China. He is currently working toward the PhD degree with the University of Hong Kong. His research interests include computer graphics and computational imaging, with a focus on integrating computer graphics techniques with computational holography to develop algorithms for next-generation display equipment.



Edmund Y. Lam (Fellow, IEEE) received the BS, MS, and PhD degrees in electrical engineering from Stanford University. He is currently a professor of electrical and electronic engineering and the Computer Engineering Program director with The University of Hong Kong. His research focuses on computational imaging algorithms, systems, and applications. He is a Founding member of The Hong Kong Young Academy of Sciences, and a fellow of Optica, SPIE, IS&T, IOP, and HKIE.

Yifan (Evan) Peng received the bachelor's and master's degrees in optical science & engineering from Zhejiang University, and the PhD degree in computer science from the University of British Columbia. He was a postdoctoral fellow with Stanford University. He is currently an assistant professor with the Department of Electrical & Electronic Engineering and with the Department of Computer Science, The University of Hong Kong. His research focuses on incorporating optical and computational techniques to enable new visual computing systems.