How to Learn a Domain-Adaptive Event Simulator?

Daxin Gu
State Key Laboratory of Virtual Reality Technology and Systems, SCSE, Beihang University, Beijing, China

Jia Li[*]
State Key Laboratory of Virtual Reality Technology and Systems, SCSE, Beihang University, Beijing, China
Peng Cheng Laboratory, Shenzhen, China

Yu Zhang[*]
State Key Laboratory of Virtual Reality Technology and Systems, SCSE, Beihang University, Beijing, China

Yonghong Tian
Department of Computer Science and Technology, Peking University, Beijing, China
Peng Cheng Laboratory, Shenzhen, China

ABSTRACT

The low-latency streams captured by event cameras have shown impressive potential in addressing vision tasks such as video reconstruction and optical flow estimation. However, these tasks often require massive training event streams, which are expensive to collect and largely bypassed by recently proposed event camera simulators. To align the statistics of synthetic events with that of target event cameras, existing simulators often need to be heuristically tuned with elaborative manual efforts and thus become incompetent to automatically adapt to various domains. To address this issue, this work proposes one of the first learning-based, domain-adaptive event simulator. Given a specific domain, the proposed simulator learns pixel-wise distributions of event contrast thresholds that, after stochastic sampling and paralleled rendering, can generate event representations well aligned with those from the data from realistic event cameras. To achieve such domain-specific alignment, we design a novel divide-and-conquer discrimination scheme that adaptively evaluates the synthetic-to-real consistency of event representations according to the local statistics of images and events. Trained with the data synthesized by the proposed simulator, the performances of state-of-the-art event-based video reconstruction and optical flow estimation approaches are boosted up to 22.9% and 2.8%, respectively. In addition, we show significantly improved domain adaptation capability over existing event simulators and tuning strategies, consistently on three real event datasets.

CCS CONCEPTS

• **Computing methodologies** → **Computer vision tasks**; *Adversarial learning*.

KEYWORDS

event camera, event simulator, adversarial learning

[*]Correspondence should be addressed to Yu Zhang and Jia Li. E-mail: zhangyulb@gmail.com, jiali@buaa.edu.cn. Website: http://cvteam.net.

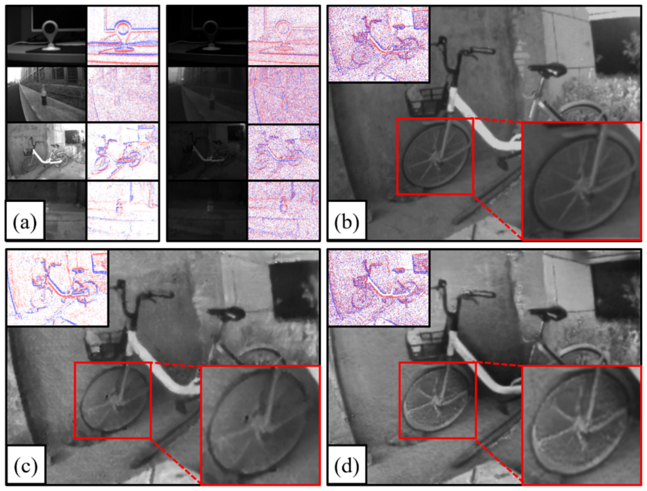

Figure 1: Motivation of the proposed approach. Noise distributions of event streams can differ sharply depending on the event cameras and their settings. For example, to capture the scene clearly in low light, the camera ISO is often set high to be light-sensitive, which, meanwhile, leads to more noise compared with normal-light photography (a). To synthesize realistic data for training low-light video reconstruction, the ESIM simulator [34] has to be tuned manually and carefully to match the target noise distribution. Otherwise, the model trained with unrealistic event streams may not generalize well on low light events (*i.e.* comparing (c) with the reference scene (b)). Our learning-based event simulator automatically adapts to the target domain without manual tuning, resulting into better trained model (d).

ACM Reference Format:
Daxin Gu, Jia Li, Yu Zhang, and Yonghong Tian. 2021. How to Learn a Domain-Adaptive Event Simulator?. In *Proceedings of the 29th ACM International Conference on Multimedia (MM ’21), October 20–24, 2021, Virtual Event, China.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3474085.3475229

1 INTRODUCTION

Event cameras, or dynamic vision sensors [23], are revolutionary visual sensors that receive wide attention in computer vision community. Their captured signals, called event streams, respond to the scene with extremely low latency and present high dynamic range, vitalizing various vision tasks such as video reconstruction [19, 37, 41, 42], image deblurring [6, 7, 32, 40] and optical flow estimation [1, 45, 50, 51]. However, training these tasks requires huge amounts of event streams and spatiotemporally aligned high-quality videos, which are usually expensive to capture due to dedicated event-to-video calibration and synchronization. Therefore, it is desired to develop efficient data synthesize techniques for event-based vision.

Recently, the ESIM simulator [34] was proposed to synthesize event streams from video sequences, mimicking the way a real event camera works. Essentially, it models several key steps of the physical generation process of events, including scene motion synthesis, contrast threshold sampling, and adaptive event rendering. Though still much simpler than the real pipeline of event-based imaging, it was shown to provide high-quality event streams trained on which, the models generalize well to processing real-world events [36, 43, 44]. The v2e simulator [9] implements a more realistic, and complicated, computational event simulation pipeline with more coverage to the ingredients of event-based imaging. However, these simulators need elaborative manual tuning of various parameters to narrow the synthetic-to-real domain gap. As shown in Fig. 1, the distribution of event streams generated by ESIM with default parameters is significantly inconsistent with that of real events, which may degenerate the performance of subsequent tasks.

To ease the manual efforts on tuning existing simulators, recent work [43] advocates a simple metric on domain alignment by measuring the *average events per pixel per second*, which is shown effective to automatically search the parameters of ESIM. However, such heuristic metric may not optimally reflect the gap between noise distributions of synthetic and real data. It was attempted by [49] on learning to generate event representations directly from video frames without using an event simulator, in which image-to-event translation was achieved via a CycleGAN [20] based framework. Nevertheless, CycleGAN training suffers from the risk of distorting the image content [10], and does not model the inherent randomness of event genreation from noisy contrast threshold sampling.

To address the aforementioned issues, we propose a novel learning-based, domain-adaptive event simulator that utilizes a small set of event streams from the target domain to automatically "calibrate" its simulation parameters. Instead of learning direct image-to-event translation as did in [49], we re-craft several essential units of ESIM, *i.e.*, contrast threshold sampling and event stream rendering, into differentiable network modules, making the simulation pipeline trainable while comply to the physics of event-based imaging. Specifically, we design deep neural networks to predict for each scene the per-pixel noise distributions of event contrast thresholds, which are sampled and used to render event streams efficiently with a reformulation of event generation process. A divide-and-conquer discrimination scheme is proposed, that adopts multiple light-weight, scene-adaptive domain discriminators for localized alignment between the generated event streams with the real ones from the target domain. Extensive experiments show that the proposed simulator surpasses previous one equipped with advanced tuning strategies. In particular, it improves the performances of the state-of-the-art approaches on event-based video reconstruction and optical flow estimation up to 22.9% and 2.8%, respectively.

In summary, the contribution of this paper are: 1) A novel learning-based event simulator that automatically "calibrates" its parameters to generate event streams from the target domain; 2) Differentiable modules for contrast threshold sampling and fast event stream rendering, which can be integrated into the framework for end-to-end training; 3) A divide-and-conquer discrimination scheme that supports localized, scene-adaptive domain alignment; 4) Outstanding performance across different datasets and subsequent tasks compared with existing simulator and tuning strategies.

2 RELATED WORK

Event simulation. Many prior works propose to simulate events from image sequences with a simplified model, by differencing the intensities of consecutive images in log-space and thresholding the differences (*e.g.*, [4, 16, 22]). However, they are limited by the frame rate of video sequences, and lack noise modeling of contrast-based threholding. ESIM [34] was proposed as the first high-fidelity event simulator that assumes continuous motion representation of the scene to improve temporal resolution of rendering and mimics the noisiness of contrast threshold sampling. It was shown to generate high-quality event streams capable of training various visual tasks, by properly tuning its simulation parameters. ESIM was further extended to combine video frame interpolation techniques to synthesize labeled event streams from real-world video datasets [12]. The v2e simulator [9] implements computational versions of more physical modules and characteristics of event camera, while also introducing far more modeling parameters to tune.

Event-based vision by learning from simulated data. Though efforts have to be spent on tuning, existing event simulators were shown to benefit various downstream vision tasks, by improving learning based methods for which large amounts of realistic training data are crucial. They greatly facilitate the advance of researches on both low-level event-based vision (*e.g.*, video reconstruction [19, 37, 41, 42], image deblurring [15, 32], video interpolation [24], optical flow estimation [1, 45, 50, 51]), high-level event-based vision (*e.g.*, object detection [29] and segmentation [2]) and 3D modeling from events (*e.g.*, [3]). However, like CMOS image sensors, event cameras present variety on low-level characteristics of captured event streams, depending on camera settings and environment. Manually tuning simulators to adapt to different event cameras can be cumbersome, necessitating automatic methods to efficiently achieve this.

Sythetic-to-real adaptation. Bridging the domain gap between the synthesized data with the real data has been extensively studied in recent years [17, 43, 49]. As for event-based vision, there are also several recent efforts on automatic and domain-adaptive event synthesis. Among them, Stoffregen *et al.* [43] propose several heuristics to tune ESIM to narrow the synthetic-to-real gap, including metrics to indicate alignment of event noise, and noise/data augmentation strategies. EventGan [49] was proposed instead as a learning approach that adopts adversarial domain alignment [33] to translate image sequences to event representations. However its generation process is deterministic, without considering the uncertainty and noise in event-based imaging. Instead of designing tuning strategies for ESIM or deterministic domain translation, we propose a

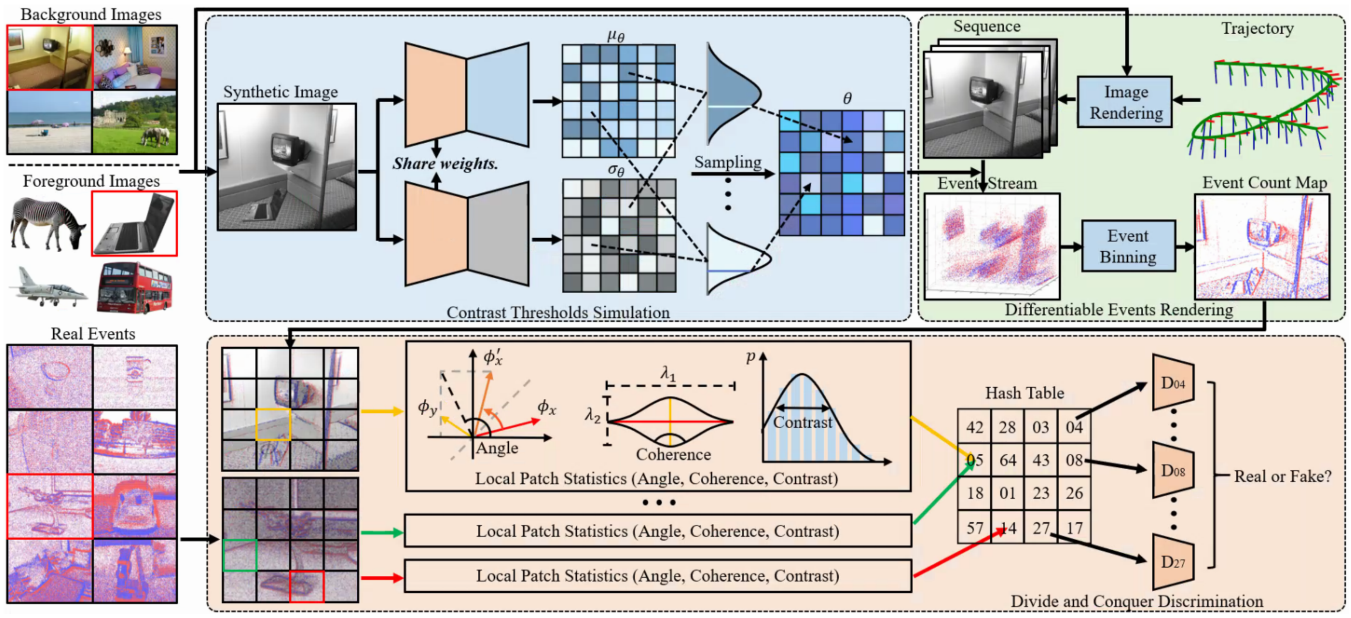

Figure 2: The pipeline of the proposed approach. Please refer to the text (Sect. 4) for details. Best viewed in electronic version.

learning-based simulator that calibrates its simulation pipeline to align with target domain characteristics. Doing so preserves the real physical formation of events, guarantees the fidelity of events in general sense, while inherits the merits of learning methods for efficient simulation parameter optimization.

3 BACKGROUND

Before diving into the the proposed approach for learning event simulation, we first briefly review the closely related and currently prevalent simulator, ESIM [34]. Essentially, to simulate event streams from a given image, it consists of the following steps:

1) Sensor trajectory generation. A motion trajectory model $\mathcal{T}$, which determines the motion dynamics of each image pixel, is generated. At a given time instant, the projected position of any image pixel can be queried from this motion model.

2) Video sequence rendering. With the trajectory, a rendering engine then generates an instant intensity image, warped from the given image(s) $\mathcal{I}_0$, at a given time instant t:

$$\mathcal{I}_t = \mathcal{R}(t; \mathcal{I}_0, \mathcal{T}). \quad (1)$$

By sampling a set of consecutive time instants $\{t_k\}_{k=1}^N$, a high frame-rate video sequence conforming to trajectory $\mathcal{T}$ can be synthesized.

3) Adaptive time sampling. To render a video sequence, a set of sampled time instants $\{t_k\}_{k=1}^N$ is needed. To improve efficiency, ESIM proposes an adaptive sampling strategy, drawing more time instants in case of fast change of the scene, while less when the scene is static. The next rendering time t_{k+1} is predicted from history based on the local scene states (*e.g.*, motion and brightness change).

4) Event stream rendering. After time instant sampling and video frame rendering, events are generated following the rule of change detection in logarithm image space [18]. Specifically, for a pixel position $\mathbf{p}$, an event is generated when the logarithm of image intensity at $\mathbf{p}$ changes up to a value called contrast threshold. This process is summarized as follows:

$$\Delta\mathcal{L}_{t_{k+1}}(\mathbf{p}) = \log(\mathcal{I}_{t_{k+1}}(\mathbf{p})) - \log(\hat{\mathcal{I}}_{t_k}(\mathbf{p})), \quad (2)$$

$$N^p_{t_{k+1}}(\mathbf{p}) = \left\lfloor \frac{h(\Delta\mathcal{L}(\mathbf{p}))}{\theta_p} \right\rfloor, \; N^n_{t_{k+1}}(\mathbf{p}) = \left\lfloor \frac{h(-\Delta\mathcal{L}(\mathbf{p}))}{\theta_n} \right\rfloor, \quad (3)$$

$$\hat{\mathcal{I}}_{t_{k+1}}(\mathbf{p}) = \hat{\mathcal{I}}_{t_k}(\mathbf{p}) \cdot \exp\left(N^p_{t_{k+1}}(\mathbf{p})\theta_p - N^n_{t_{k+1}}(\mathbf{p})\theta_n\right), \quad (4)$$

where $h(\cdot) = \max(\cdot, 0)$ is the half-rectify function, θ_p and θ_n are contrast thresholds for positive and negative polarity, respectively. Note that instead of differencing the instant image $\mathcal{I}_{t_{k+1}}$ with $\mathcal{I}_{t_k}$ in (2), it is differenced with the intermediate image $\hat{\mathcal{I}}_{t_k}$ which accumulates the historically generated events and is updated continuously, to meet the real physical process of event generation [9].

The contrast thresholds θ were observed not constant but follow normal distributions [23]. To mimic this, ESIM manually defines such distribution, and samples specific values during each generation step (3). A number of events (*i.e.*, $N^p_{t_{k+1}}(\mathbf{p})$ or $N^n_{t_{k+1}}(\mathbf{n})$) can be jointly generated by floor rounding the division between changed log-intensity and threshold (3).

4 DOMAIN-ADAPTIVE EVENT SIMULATOR

Instead of manual tuning of simulation parameters, we propose a learning-based event simulation framework, as summarized in Fig. 2. It consists of the essential components as defined in ESIM, but replaced with differentiable versions. Like ESIM, the proposed framework creates synthesized training scenes by warping randomly sampled high-quality images with predefined trajectories. Different from ESIM, however, the proposed *contrast threshold simulation* module adopts neural networks to automatically estimate the distribution parameters, $\boldsymbol{\mu}_\theta$ and $\boldsymbol{\sigma}_\theta$, of the spatially adaptive event contrast thresholds, from the synthesized scene. Event contrast thresholds are sampled from the learned distributions, by which event streams are generated from the scene sequences with the *differentiable event rendering* module.

To guarantee the generated event streams fall within the domain of the target dataset, a domain alignment scheme is proposed. To this end the synthesized event streams are binned to generate *event count maps* [26, 28, 35], which, with those generated from the real event streams from the target dataset, are fed into a *divide-and-conquer domain discrimination* scheme. Local patches are extracted from the event count map, for each of which structure is analysed and used to map it to a particular discriminator handling that structure. In the rest, we elaborate these techniques in more details.

4.1 Training Video Sequence Generation

We follow Stoffregen *et al.* [43] to sample scene trajectories and render training video sequences, which we briefly summarize for completeness. To render a scene, one or multiple foreground images and a background image are first sampled from a large dataset, for which we use the *unlabeled* subset of the MS COCO dataset [25]. For foreground images, we adopt the annotated instance masks to filter out the background. Each foreground or background image is associated with a synthetic trajectory, formed by affine transformations including translation, scaling and rotation, and a velocity. Our video sequence renderer $\mathcal{R}$, taking similar form with (1), warps a small set of foreground objects and the background image by their respective motion trajectories with different levels of velocities, and composed to produce the rendered image.

4.2 Learning Event Stream Rendering

Given the video sequence renderer $\mathcal{R}$, synthesizing event streams, as discussed in Sect. 3, requires 1) defining a sampling strategy of rendering time, and 2) synthesizing the events from the image rendering results at the sampled time instants.

We follow a strategy proposed in ESIM to sequentially sample the rendering time, based on the state of the scene motion. Specifically, the next rendering time t_{k+1} is predicted from the maximal velocity magnitude over all positions $\mathbf{p}$ at time t_k:

$$t_{k+1} = t_k + \max\left(\lambda_t \frac{1}{\max_{\mathbf{p}} |v_{t_k}(\mathbf{p})|}, \delta\right), \tag{5}$$

where $\lambda_t = 0.5$ trade-offs the rendering accuracy and speed, δ models the refractory period of event camera and is set to 1ms.

Our event stream rendering process is similar with the general procedures for event simulation from images (2) ~ (4), in which contrast thresholds of positive and negative polarity, namely θ_p and θ_n, are first sampled. Unlike previous simulators, contrast thresholds are automatically learned in our approach, which will be explained shortly after for conciseness. With the thresholds, (2) ~ (4) synthesize events sequentially, by differencing the latest rendered image $\mathcal{I}_{t_{k+1}}$ with a dynamically updated history image $\hat{\mathcal{I}}_{t_k}$. However, such a generation procedure, if unrolled, involves recursive gradient updates on the learned thresholds θ (see (3) and (4)), causing gradient instability for long-time rendering during training.

Near-parallel reformulation of event rendering. We propose an approximation of event generation (2) ~ (4), enabling nearly paralleled computation of event counts $N^p_{t_k}$ and $N^n_{t_k}$ of different time intervals among consecutive t_ks. Specifically, it can be proved that under the mild assumption $\theta_p \approx \theta_n = \theta$, the event counts $N^p_{t_{k+1}}$ and $N^n_{t_{k+1}}$ take the following formula:

$$\begin{aligned} &N^p_{t_{k+1}}(\mathbf{p}) = h(D_{t_{k+1}}(\mathbf{p})),\ N^n_{t_{k+1}}(\mathbf{p}) = h(-D_{t_{k+1}}(\mathbf{p})), \\ &\text{where } D_{t_{k+1}}(\mathbf{p}) = S_{t_{k+1}}(\mathbf{p}) - S_{t_k}(\mathbf{p}). \end{aligned} \tag{6}$$

Here, $S_{t_{k+1}}$ represents the accumulated change of polarities:

$$\begin{aligned} S_{t_{k+1}}(\mathbf{p}) &= \lfloor \hat{S}_{t_{k+1}}(\mathbf{p}) \rfloor + r_{t_{k+1}}(\mathbf{p}), \\ \text{where } \hat{S}_{t_{k+1}}(\mathbf{p}) &= \frac{\log(\mathcal{I}_{t_{k+1}}(\mathbf{p})) - \log(\hat{\mathcal{I}}_{t_0}(\mathbf{p}))}{\theta(\mathbf{p})}, \end{aligned} \tag{7}$$

and $r_{t_{k+1}}$ is an adaptive residual to compensate the rounding error. If $\hat{\mathcal{S}}_{t_{k+1}} = \lfloor \hat{\mathcal{S}}_{t_{k+1}} \rfloor$, *i.e.* is an integer, then $r_{t_{k+1}} = 0$. Otherwise,

$$r_{t_{k+1}}(\mathbf{p}) = \begin{cases} -1, & \lfloor \hat{S}_{t_{k+1}}(\mathbf{p}) \rfloor > \lfloor \hat{S}_{t_k}(\mathbf{p}) \rfloor \wedge \hat{S}_{t_{k+1}}(\mathbf{p}) < 0 \\ 1, & \lfloor \hat{S}_{t_{k+1}}(\mathbf{p}) \rfloor < \lfloor \hat{S}_{t_k}(\mathbf{p}) \rfloor \wedge \hat{S}_{t_{k+1}}(\mathbf{p}) > 0 \\ r_{t_k}(\mathbf{p}), & \lfloor \hat{S}_{t_{k+1}}(\mathbf{p}) \rfloor = \lfloor \hat{S}_{t_k}(\mathbf{p}) \rfloor \\ 0, & \text{otherwise.} \end{cases} \tag{8}$$

From (8), $\hat{S}_{t_k}$ can be easily computed in parallel for every t_k, which unnecessites recursive gradient propagation for θ. Computing r_{t_k}, though not paralleled, does not affect gradient calculation for θ. In practice we find it result into more stable training and much faster convergence. Note that the assumption $\theta_p \approx \theta_n = \theta$ is mild: they are often similar in event cameras, roughly complying to $\theta_p \approx \theta_n \cdot \mathcal{N}(1.0, 0.1)$ where $\mathcal{N}(\cdot,\cdot)$ is normal distribution [43].

The contrast thresholds θ define the uncertainty of event generation, which previous simulators [9, 34] sample from a manually set, globally uniform distribution. However, like the CMOS image sensor noise, event contrast threshold noise is inherently non-uniform and spatially varying, depending on the camera settings, the environment illumination, and temperature [31]. Unfortunately, it is still difficult to accurately calibrate such noise distributions [11], leave behind manual tuning to simulate them. In this work, we rely on neural networks to address such difficulty.

Contrast threshold learning. Our contrast threshold learning network takes as input a rendered image of the scene (*i.e.*, the middle image of the full rendered sequence), and outputs per-pixel distributions of threshold noise. We follow previous observations that contrast thresholds are mostly normally distributed [23], and model such distributions with two pixel-varing maps $\boldsymbol{\mu}_\theta = \mu_\theta(\cdot)$ and $\boldsymbol{\sigma}_\theta = \sigma_\theta(\cdot)$, representing the sufficient statistics, *i.e.*, mean and variance of normal distribution, respectively. The network is of UNet structure [39], using residual blocks in its encoder, and skip connections between them. We refer the readers to our supplementary for more details of the network. Since that the variance is always positive, we compute exponential of the network output to get variance $\boldsymbol{\sigma}_\theta$.

Generating the contrast thresholds θ requires sampling from $\boldsymbol{\mu}_\theta$ and $\boldsymbol{\sigma}_\theta$. This is achieved with the *reparameterization trick* [21], by firstly sampling a variable from a standard nominal $z(\mathbf{p}) \sim \mathcal{N}(0, 1)$, then scaling and shifting it via $\theta(\mathbf{p}) = \mu_\theta(\mathbf{p}) + z(\mathbf{p}) \cdot \sigma_\theta(\mathbf{p})$. The generated thresholds are then clipped below with a small constant to ensure positiveness, $\theta(\mathbf{p}) = \max(\theta(\mathbf{p}), \epsilon)$, where $\epsilon = 0.01$.

With the sampled per-pixel thresholds, we adopt (6) to generate event streams, and then feed them with the real events into the

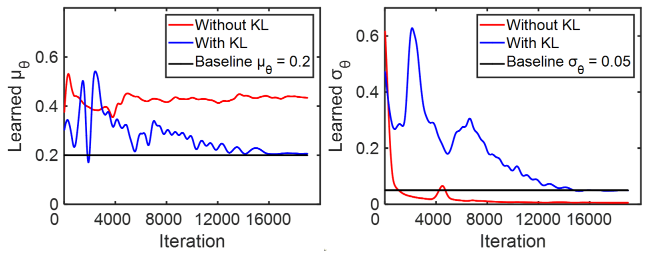

Figure 3: Illustrating KL-regularization. Two models of our approach, with and without KL-regularizations, are trained to adapt to a synthetic dataset with threshold distribution in the form $\mathcal{N}(0.2, 0.05)$. With KL-regularization, the target distribution is successfully learned.

domain discriminator for domain adaptation, which provides training signals to contrast thresholds. However, simply doing this the learned variance often diminish to zeros, as shown in Fig. 3. Due to the lack of direct supervision to threshold distribution learning, it is prone to fall into bad local minima without regularization.

We find this issue can be addressed by regularizing the learned distributions close to a prior, via minimizing their KL-divergence:

$$L_r(\mathbf{p}) = -\lambda_r D_{KL}\left(\mathcal{N}(\mu_\theta(\mathbf{p}), \sigma_\theta(\mathbf{p}))||\mathcal{N}(0,1)\right), \tag{9}$$

where λ_r controls the strength of regularization, which can be set large and annealed to zero during training, to get rid of bad minima at early stage. In practice, setting λ_r to a small constant 10^{-5} without annealing already works well. As illustrated in Fig. 3, this regularization is crucial to make distribution learning succeed.

4.3 Divide-and-Conquer Discrimination

The generated event streams are compared with the real ones randomly sampled from the target dataset for domain discrimination. To represent the event streams we use stacked event representations [28], binning them into a single image by pixel-wisely counting the number of events occurred in a small time window (*i.e.*, 50ms). We separate the counting of positive and negative events, resulting into two event count maps, $\mathbf{M}_g^p$ and $\mathbf{M}_g^n$.

We follow the recent advance on adversarial learning for domain adaptation [14, 47, 52], training the generated event count maps to fool a domain discriminator:

$$L_D = \mathcal{E}_{\mathbb{M}_r \sim P_r(\mathbb{M}_r)}[D(\mathbb{M}_r)] - \mathcal{E}_{\mathbb{M}_g \sim P_g(\mathbb{M}_g)}[D(\mathbb{M}_g)], \tag{10}$$

where $\mathcal{E}$ is empirical mean, $\mathbb{M}_g = \{\mathbf{M}_g^p, \mathbf{M}_g^n\}$ is the set of generated event count maps, and P_g is the learned distribution of them. $\mathbb{M}_r$ and P_r are defined analogously for real data. (10) is the objective of Wasserstein GAN [13], which is maximized for the discriminator D, and minimized for the threshold learning network.

The discriminator $D(\cdot)$, in most previous works, is implemented with a single, strong network expected to learn all characteristics of real domain. However, it does not work well in our case for a number of reasons. First, GANs are known prone to learn only limited modes of data distributions, other than full of them [8]. Therefore, it is expected that local regions of the scene are not treated in balanced way, with only a few ones (*e.g.*, those with sharp domain discrepancy) being attended more extensively than others. Second, generation of events is inherently a low-level process, depending on local scene statistics such as edges and velocities. Learning a high-level representation with large context is not necessary, while easier to cause overfitting due to the more complexities to learn.

Keeping this in mind, we propose to adopt multiple, localized domain discriminators adapted to different types of local scene structures. To this end the generated event count maps are partitioned into small, fixed-size local patches. For each local patch, we analyse the type of its structure, and map it to a particular domain discriminator handling that structure. Specifically,

$$D(\mathcal{P}) = \sum_k \mathbb{1}(s(\mathcal{P}) = k) D_k(\mathcal{P}), \tag{11}$$

where $s(\cdot)$ is a hashing function that analyses structure of a patch $\mathcal{P}$ and maps it to the index of the corresponding discriminator.

To efficiently analyse structure of a local patch, we follow [38] that performs eigen analysis of patch gradients and extract three attributes: edge orientation, coherence and strength. Such attributes are tightly related to intensity contrast across image edges, conceptually aligning with the factors that affect physical event generation. We compute them on the patches extracted from the rendered image scene aligned in time with the generated event count maps. For real events, such image can be obtained in free if a hybrid event camera is used (e.g., [5]), or reconstructed from events [37].

We only keep the orientation and coherence values while discard the strengths, since intensity strength is not domain invariant in image space due to dependence on camera-related factors (*e.g.*, dynamic range, sensor blur) or image compression. Instead, we directly measure the contrast strength in event space:

$$c(\mathbf{m}) = \sum_{\mathbf{p} \in \mathbb{P}(\mathbf{m})} \sum_{\mathbf{q} \in \mathbb{N}(\mathbf{p})} ||\nabla_{\mathbf{m}}(\mathbf{p}, \mathbf{q})||^2 P_{\nabla_{\mathbf{m}}}(\mathbf{p}, \mathbf{q}), \tag{12}$$

where $\mathbb{P}(\mathbf{m})$ is the set of local positions of patch $\mathbf{m}$, $\mathbb{N}(\mathbf{q})$ is the set of neighbour positions of $\mathbf{p}$ in 4-neighbour system, and $\nabla(\cdot)$ denotes the operator of gradient magnitude. $P_{\nabla_{\mathbf{m}}}$ represents the density of gradient magnitude values, evaluated within patch $\mathbf{m}$ by histogram binning. Intuitively, (4) signifies the "focusness" of events, whose value is large if most events are cluttered to a few structured positions (*e.g.*, strong edges).

We quantize the orientation, coherence and contrast values by 24, 3, 3, resulting into 216 local discriminators. After extracting the three values for a patch, it can be thus mapped to a particular discriminator handling that type. Each discriminator is a small MLP that consumes the vectorized version of four patch images (positive and negative event count maps, as well as their gradient magnitudes), and produce 256, 128, 1 channels.

Discriminator gradient weighting. Since patch distributions of natural images are not uniform but long-tail [38], there exists certain structure types rarely sampled. The corresponding discriminators will be thus insufficiently trained, providing incorrect gradients to threshold learning. We take this into account, weighting the gradients of different discriminators based on real patch distribution. Specifically, the weight for the kth discriminator is

$$\omega_k = \alpha \cdot \frac{1}{K} + (1-\alpha) \cdot \frac{S_k}{\sum_k S_k}, \tag{13}$$

Table 1: Benchmarking video reconstruction performance trained with different event simulation pipelines on the IJRR [30], MVSEC [48] and HQF [43] datasets. Top place highlighted in bold.

Dataset	IJRR [30]				MVSEC [48]				HQF [43]			
Method	E2V [36]	S2R [43]	E-GAN [49]	Ours	E2V [36]	S2R [43]	E-GAN [49]	Ours	E2V [36]	S2R [43]	E-GAN [49]	Ours
MSE↓	0.07	0.03	0.06	**0.02**	0.29	0.11	0.14	**0.08**	0.14	**0.03**	0.05	0.04
SSIM↑	0.61	0.64	0.56	**0.68**	0.27	0.35	0.34	**0.43**	0.46	0.58	0.62	**0.64**
LPIPS↓	0.28	**0.22**	0.24	**0.22**	0.65	**0.47**	0.52	**0.47**	0.46	**0.26**	0.30	**0.26**

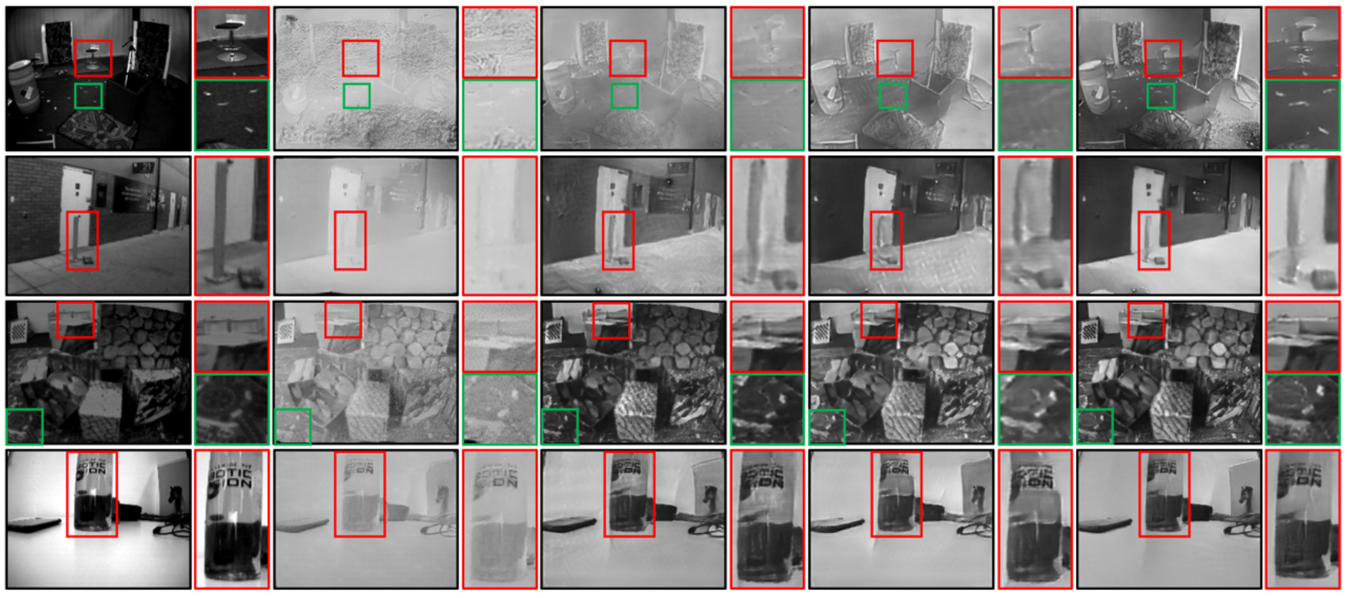

Figure 4: Representative results generated by different approaches. Note that there exists visual gaps between the reconstructed color with that of reference images, since models are trained on synthetic COCO sequences instead of images from the target domain. Best viewed with zoom.

where K is the number of discriminators, S_k is the number of patches sampled for the kth structure during training. The tradeoff parameter α is set to 0.5 to account for distribution modeling error.

5 EXPERIMENT

5.1 Experimental Protocols

Since the proposed approach aims to generate realistic data to improve model training in downstream tasks, we follow the evaluation protocols of recent work [43], which sets the save objective with ours, and evaluate our approach on two tasks: event-based video reconstruction and optical flow estimation.

Event-to-video reconstruction. We choose the representative E2V [36] network as the baseline model. As in [43], we train E2V using the synthetic events from our simulator, and evaluate its reconstruction performance on three datasets: IJRR [30], MVSEC [48] and HQF [43]. These datasets are all captured with real-world event cameras, each containing minutes to hours of event streams with corresponding reference video frames captured synchronously. On these datasets we apply three metrics, Mean Square Error (MSE), Structural Similarity Index (SSIM), and the perceptual metric LPIPS [46], to quantify the video reconstruction quality.

Optical flow estimation. The event-based flow estimation network EV-FlowNet [50] is chosen as baseline model. Again, we evaluate its performance on the same three datasets IJRR, MVSEC and HQF. Since that groundtruth optical flows are absent in IJRR and HQF, we follow [43] and use the Flow Warp Loss (FWL), which quantifies the photometric error by optical warping between consecutive frames as a surrogate metric of optical flow quality. We also report the flow-based metric Average Endpoint Error (AEE) on MVSEC as it provides groundtruth optical flows.

Benchmarking methods. To the best of our knowledge, S2R [43] and E-GAN [49] are the only methods in literature that aim to improve event simulation. We directly compare with the reported results of S2R on the same datasets and downstream tasks. For E-GAN, we apply the released model pretrained on the MVSEC dataset on the synthesized video sequences created from MS COCO images to generate event streams, on which the downstream task

Table 2: Benchmarking optical flow estimation performance trained with different event simulation pipelines on the IJRR [30], MVSEC [48] and HQF [43] datasets. Best highlighted in bold.

Dataset	IJRR [30]				MVSEC [48]				HQF [43]			
Method	EVFlow [50]	S2R [43]	E-GAN [49]	Ours	EVFlow [50]	S2R [43]	E-GAN [49]	Ours	EVFlow [50]	S2R [43]	E-GAN [49]	Ours
FWL$\uparrow$	1.32	1.45	1.43	**1.49**	1.12	1.30	1.32	**1.34**	1.20	1.35	1.45	**1.48**

Table 3: Comparing optical flow estimation results on the sequences from MVSEC [48], following the same evaluation settings of [43]. Best highlighted in bold.

Sequence	outdoor_day1		outdoor_day2		indoor_flying1		indoor_flying2		indoor_flying3	
	AEE	$\%_{\text{Outlier}}$	AEE	$\%_{\text{Outlier}}$	AEE	$\%_{\text{Outlier}}$	AEE	$\%_{\text{Outlier}}$	AEE	$\%_{\text{Outlier}}$
Zeros	4.31	0.39	1.07	**0.91**	1.10	**1.00**	1.74	**0.89**	1.50	**0.94**
EVFlow*	0.49	**0.20**	–	–	1.03	2.20	1.72	15.10	1.53	11.90
S2R	0.68	0.99	0.82	0.96	0.56	**1.00**	0.66	1.00	0.59	1.00
E-GAN	0.59	1.00	0.69	0.97	0.60	**1.00**	0.75	1.00	0.64	1.00
Ours	**0.53**	1.00	**0.68**	0.97	**0.49**	**1.00**	**0.64**	0.99	**0.54**	1.00

* The baseline EVFlow is trained on the sequence *outdoor_day2*, leading to the absence of evaluation on it.

Table 4: Ablation analysis of the proposed divide-and-conquer domain discrimination scheme. See text for details.

Setting	KL-div.	μ_θ diff.	σ_θ diff.
w/o divide-and-conquer	5.46	0.14 ± 0.22	0.14 ± 0.25
w/o re-weighting	1.87	0.03 ± 0.04	0.04 ± 0.06
full model	**1.50**	**0.01 ± 0.04**	**0.02 ± 0.04**

Table 5: Analysing patch structure attributes for domain adaptation. See text for details.

Angle	Contrast	Coherence	KL-div.	μ_θ diff.	σ_θ diff.
			2.68	0.04 ± 0.04	0.11 ± 0.06
✓			2.64	0.05 ± 0.04	0.07 ± 0.06
	✓		10.80	0.06 ± 0.04	0.17 ± 0.10
		✓	7.23	0.01 ± 0.10	0.09 ± 0.09
✓	✓		1.61	0.03 ± 0.05	0.04 ± 0.05
✓		✓	1.53	0.03 ± 0.03	0.04 ± 0.03
	✓	✓	2.03	0.01 ± 0.05	0.05 ± 0.05
✓	✓	✓	**1.50**	**0.01 ± 0.04**	**0.02 ± 0.04**

networks are trained. The trained E2V and EVFlow models are served as strong baselines.

Implementation details. During training, we randomly select images from the COCO dataset and real event streams from the target event dataset. We render 5620 sequences using COCO images, each contains 10 frames, with $0 \sim 30$ foreground objects. We generate event streams of 1/30 second and sample real event streams with same time length. We train our simulator using AdaBelief [53] optimizer for 200 epochs, using initial learning rate 4e-4 with warmup strategy at the beginning, gradually decayed to zero by cosine annealing [27]. Training is distributed on 4 GTX1080TI GPUs, each holding a batch of 12 sequences. We train E2V and EVFlow following the same settings of [43].

5.2 Comparisons on Downstream Tasks

Tab. 1 summarizes the quantitative comparison results on event-based video reconstruction. The proposed approach achieves the leading place in nearly all the metrics on all the datasets. Particularly, it shows large improvements on SSIM, indicating that our approach possesses the minimal distoration of scene structures. In Fig. 4, we illustrate representative video reconstruction results generated by different approaches from the same sets of event streams. While the original E2V does not generate plausible details and local contrast, S2R and E-GAN presents distorted local details. Trained with events synthesized by the proposed simulator, the best video reconstruction quality is achieved, demonstrating the effectiveness of better domain-adapted training data.

The optical flow estimation results trained with different data simulation pipelines are summarized in Tab. 2 and Tab. 3. Our approach achieves the best performance (note that for the FWL metric defined in [43], larger indicates better). From Tab. 3, the proposed approach indeed benefits practical flow estimation through evaluations with groundtruth real-world flows.

Specifically, the proposed simulator boosts the state-of-the-art performances on event-based video reconstruction and optical flow estimation up to 22.9% and 2.8%, respectively. More visual comparisons and the per-sequence quantitative results are referred to our supplementary material.

5.3 Performance Analysis

In this section, we look into how the proposed framework works, via a series of dedicated experiments.

Ablation study of divide-and-conquer discrimination. First, we justify the design choices of the proposed divide-and-conquer domain adaptation. We design a synthetic experiment using a subset of virtually rendered video sequences to generate event streams with a globally uniform contrast threshold distribution ($\mu_\theta = 0.2$, $\sigma_\theta = 0.05$), which is treated as the target domain. The training task is thus to align the learned threshold distributions with the

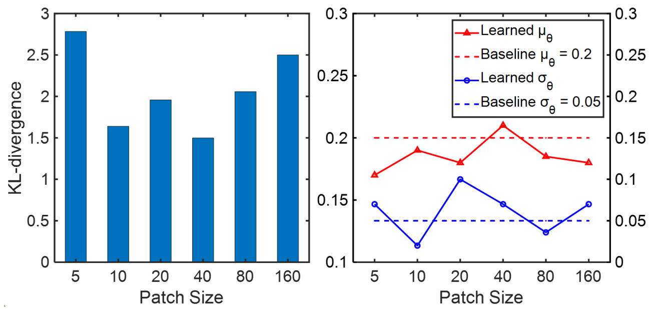

Figure 5: Domain alignment performance as a function of patch size in domain discrimination. See text for details.

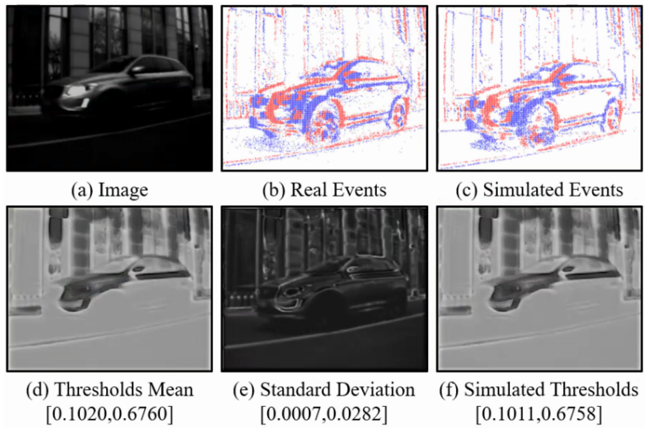
(a) Image
(b) Real Events
(c) Simulated Events
(d) Thresholds Mean [0.1020,0.6760]
(e) Standard Deviation [0.0007,0.0282]
(f) Simulated Thresholds [0.1011,0.6758]

Figure 6: Visualizing learned contrast threshold distributions. See text for details. Best viewed in color.

synthetic one, which can be quantitatively evaluated by measuring the KL-divergence between the predictive and "groundtruth" distributions, or the absolute difference between the distribution variables. In Tab. 4, *w/o divide-and-conquer* represents the variant with only global domain alignment with a single discriminator, while *w/o re-weighting* excludes the discriminator re-weighting scheme. Clearly these variants lead to drop of performance, demonstrating the effectiveness of these designs.

Analysing the contributions of structural patch attributes. In Tab. 5, we summarize the results by isolating one or more types of patch structure attributes in the discriminator division, under the same synthetic experimental setting. It shows that isolating any attribute harms the final result, suggesting that these attributes have complementary effect. It also shows that angle and contrast are the most dominant attributes, as excluding any of them makes the result degenerate significantly.

We further analyse the impact of different patch sizes for domain discrimination. In Fig. 5, we illustrate the distribution gaps as a function of patch size. As discussed earlier, too large patch size, though rich context, will not improve the performance for event stream simulation. Empirically we find patches with 40 pixels in each border perform best in our experiments.

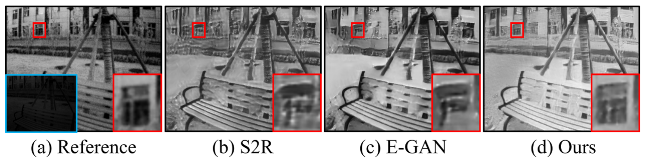

Figure 7: Comparisons on domain adaptation capability of different simulation pipelines in real dark light scenario, using video reconstruction as downstream task.

Visualizing the learned contrast thresholds. To see what threshold maps are generated by the proposed approach, we visualize one such example on simulating real-world events in Fig. 6. We train the simulator to generate adapted event streams that are aligned with the real ones captured from an outdoor scene. The simulator correctly generates similar event representations (comparing (b) and (c)). Meanwhile, the contrast distribution maps, as shown in (d) and (e), present clear edge-aware effect. Note that this somewhat deviates from real event distributions, as event cameras are only aware of change of illumination but not real scene structures. This may be caused by the simplification of event contrast modeling, which is remedied with the emphasis of scene edges during the learning process.

Visual comparisons on real low-light scenes. Generally, low-light scenes are difficult to handle in computational photography since that the starving light causes large noise variance and complicates the noise structure. To illustrate the advantage of the proposed approach in this challenging scenario, we capture 20 sequences of real-world low light scenes, each lasts for about 2 minutes. We train different approaches to adapt to this dataset, compare the performance visually in Fig. 7 (since that groundtruth clean images are difficult to collect in low light). The results show that the proposed approach generates cleaner results with less noise and improved details. More results can be found in our supplementary material.

6 CONCLUSION

In this paper, we propose a novel learning-based event camera simulator that automatically adapts to a target domain of real-world event streams. To achieve this, we propose learnable modules of contrast threshold modeling/sampling and efficient event stream rendering. A divide-and-conquer domain alignment scheme is further proposed to transfer the target-domain characteristics to the simulator in content-adaptive, localized manner. The proposed approach improves the state-of-the-art performances of event-based video reconstruction and optical flow estimation significantly, by training models on the event streams synthesized from it.

Through this work, we would like to ease the manual efforts on tuning event camera simulators, as well as open a way of learning event stream formation in principled manner. Our future interest lies in learning task-specific event stream simulation, benefiting the ultimate task performance more straightforwardly with simulation policy searching techniques (*e.g.*, reinforcement learning).

7 ACKNOWLEDGMENTS

This work was supported in part by National Natural Science Foundation of China (No. 61922006, No. 61825101, and No. 62027804).

REFERENCES

[1] Mohammed Almatrafi and Keigo Hirakawa. 2020. DAViS Camera Optical Flow. *IEEE Transactions on Computational Imaging* 6 (2020), 396–407.

[2] Iñigo Alonso and Ana C. Murillo. 2019. EV-SegNet: Semantic Segmentation for Event-Based Cameras. In *CVPR*. 1624–1633.

[3] Alexis Baudron, Zihao W. Wang, Oliver Cossairt, and Aggelos K. Katsaggelos. 2020. E3D: Event-Based 3D Shape Reconstruction. *Computing Research Repository (CoRR)* (2020).

[4] Yin Bi and Yiannis Andreopoulos. 2017. PIX2NVS: Parameterized conversion of pixel-domain video frames to neuromorphic vision streams. In *IEEE Conference on Image Processing (ICIP)*. 1990–1994.

[5] Christian Brandli, Raphael Berner, Minhao Yang, Shih-Chii Liu, and Tobi Delbrück. 2014. A 240 × 180 130 dB 3 μs Latency Global Shutter Spatiotemporal Vision Sensor. *IEEE Journal of Solid State Circuits* 49, 10 (2014), 2333–2341.

[6] Christian Brandli, Lorenz Müller, and Tobi Delbrück. 2014. Real-time, high-speed video decompression using a frame- and event-based DAVIS sensor. In *IEEE International Symposium on Circuits and Systemss, ISCAS*. 686–689.

[7] Haoyu Chen, Minggui Teng, Boxin Shi, Yizhou Wang, and Tiejun Huang. 2020. Learning to Deblur and Generate High Frame Rate Video with an Event Camera. *Computing Research Repository (CoRR)* (2020).

[8] Antonia Creswell, Tom White, Vincent Dumoulin, Kai Arulkumaran, Biswa Sengupta, and Anil A. Bharath. 2018. Generative Adversarial Networks: An Overview. *IEEE Signal Processing Magazine* 35, 1 (2018), 53–65.

[9] Tobi Delbrück, Yuhuang Hu, and Zhe He. 2020. V2E: From video frames to realistic DVS event camera streams. *Computing Research Repository (CoRR)* (2020).

[10] Huan Fu, Mingming Gong, Chaohui Wang, Kayhan Batmanghelich, Kun Zhang, and Dacheng Tao. 2019. Geometry-Consistent Generative Adversarial Networks for One-Sided Unsupervised Domain Mapping. In *CVPR*. 2427–2436.

[11] Guillermo Gallego, Tobi Delbrück, Garrick Orchard, Chiara Bartolozzi, Brian Taba, Andrea Censi, Stefan Leutenegger, Andrew J. Davison, Jörg Conradt, Kostas Daniilidis, and Davide Scaramuzza. 2019. Event-based Vision: A Survey. *Computing Research Repository (CoRR)* (2019).

[12] Daniel Gehrig, Mathias Gehrig, Javier Hidalgo-Carrió, and Davide Scaramuzza. 2020. Video to Events: Recycling Video Datasets for Event Cameras. In *CVPR*. 3583–3592.

[13] Ishaan Gulrajani, Faruk Ahmed, Martín Arjovsky, Vincent Dumoulin, and Aaron C. Courville. 2017. Improved Training of Wasserstein GANs. In *Advances in Neural Information Processing Systems (NeurIPS)*. 5767–5777.

[14] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A. Efros. 2017. Image-to-Image Translation with Conditional Adversarial Networks. In *CVPR*. 5967–5976.

[15] Zhe Jiang, Yu Zhang, Dongqing Zou, Jimmy S. J. Ren, Jiancheng Lv, and Yebin Liu. 2020. Learning Event-Based Motion Deblurring. In *CVPR*. 3317–3326.

[16] Jacques Kaiser, Juan Camilo Vasquez Tieck, Christian Hubschneider, Peter Wolf, Michael Weber, Michael Hoff, Alexander Friedrich, Konrad Wojtasik, Arne Rönnau, Ralf Kohlhaas, Rüdiger Dillmann, and Johann Marius Zöllner. 2016. Towards a framework for end-to-end control of a simulated vehicle with spiking neural networks. In *IEEE International Conference on Simulation, Modeling, and Programming for Autonomous Robots (SIMPAR)*. 127–134.

[17] Amlan Kar, Aayush Prakash, Ming-Yu Liu, Eric Cameracci, Justin Yuan, Matt Rusiniak, David Acuna, Antonio Torralba, and Sanja Fidler. 2019. Meta-Sim: Learning to Generate Synthetic Datasets. In *ICCV*. 4550–4559.

[18] Matthew L. Katz, Konstantin Nikolic, and Tobi Delbrück. 2012. Live demonstration: Behavioural emulation of event-based vision sensors. In *IEEE International Symposium on Circuits and Systems (ISCAS)*. 736–740.

[19] Hanme Kim, Ankur Handa, Ryad Benosman, Sio-Hoi Ieng, and Andrew J. Davison. 2014. Simultaneous Mosaicing and Tracking with an Event Camera. In *BMVC*.

[20] Taeksoo Kim, Moonsu Cha, Hyunsoo Kim, Jung Kwon Lee, and Jiwon Kim. 2017. Learning to Discover Cross-Domain Relations with Generative Adversarial Networks. In *ICML*, Vol. 70. 1857–1865.

[21] Diederik P. Kingma and Max Welling. 2014. Auto-Encoding Variational Bayes. In *International Conference on Learning Representations (ICLR)*.

[22] Nathan P. Koenig and Andrew Howard. 2004. Design and use paradigms for Gazebo, an open-source multi-robot simulator. In *IEEE International Conference on Intelligent Robots and Systems*. 2149–2154.

[23] Patrick Lichtsteiner, Christoph Posch, and Tobi Delbrück. 2008. A 128×128 120 dB 15 μs Latency Asynchronous Temporal Contrast Vision Sensor. *IEEE Journal of Solid State Circuits* 43, 2 (2008), 566–576.

[24] Songnan Lin, Jiawei Zhang, Jinshan Pan, Zhe Jiang, Dongqing Zou, Yongtian Wang, Jing Chen, and Jimmy S. J. Ren. 2020. Learning Event-Driven Video Deblurring and Interpolation. In *ECCV*, Vol. 12353. 695–710.

[25] Tsung-Yi Lin, Michael Maire, Serge J. Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. 2014. Microsoft COCO: Common Objects in Context. In *ECCV*, Vol. 8693. 740–755.

[26] Min Liu and Tobi Delbrück. 2018. Adaptive Time-Slice Block-Matching Optical Flow Algorithm for Dynamic Vision Sensors. In *BMVC*. 88.

[27] Ilya Loshchilov and Frank Hutter. 2017. SGDR: Stochastic Gradient Descent with Warm Restarts. In *International Conference on Learning Representations (ICLR)*.

[28] Ana I. Maqueda, Antonio Loquercio, Guillermo Gallego, Narciso García, and Davide Scaramuzza. 2018. Event-Based Vision Meets Deep Learning on Steering Prediction for Self-Driving Cars. In *CVPR*. 5419–5427.

[29] Anton Mitrokhin, Cornelia Fermüller, Chethan Parameshwara, and Yiannis Aloimonos. 2018. Event-Based Moving Object Detection and Tracking. In *IEEE International Conference on Intelligent Robots and Systems (IROS)*. 1–9.

[30] Elias Mueggler, Henri Rebecq, Guillermo Gallego, Tobi Delbrück, and Davide Scaramuzza. 2017. The event-camera dataset and simulator: Event-based data for pose estimation, visual odometry, and SLAM. *International Journal of Robotics Research* 36, 2 (2017), 142–149.

[31] Y. Nozaki and T. Delbruck. 2017. Temperature and Parasitic Photocurrent Effects in Dynamic Vision Sensors. *IEEE Transactions on Electron Devices* 8 (2017), 1–7.

[32] Liyuan Pan, Cedric Scheerlinck, Xin Yu, Richard Hartley, Miaomiao Liu, and Yuchao Dai. 2019. Bringing a Blurry Frame Alive at High Frame-Rate With an Event Camera. In *CVPR*. 6820–6829.

[33] Sinno Jialin Pan and Qiang Yang. 2010. A Survey on Transfer Learning. *IEEE Transactions Knowledge and Data Engineering* 22, 10 (2010), 1345–1359.

[34] Henri Rebecq, Daniel Gehrig, and Davide Scaramuzza. 2018. ESIM: an Open Event Camera Simulator. In *Conference on Robot Learning*, Vol. 87. 969–982.

[35] Henri Rebecq, Timo Horstschaefer, Guillermo Gallego, and Davide Scaramuzza. 2017. EVO: A Geometric Approach to Event-Based 6-DOF Parallel Tracking and Mapping in Real Time. *IEEE Robotics and Automation Letters* 2, 2 (2017), 593–600.

[36] Henri Rebecq, René Ranftl, Vladlen Koltun, and Davide Scaramuzza. 2019. Events-To-Video: Bringing Modern Computer Vision to Event Cameras. In *CVPR*. 3857–3866.

[37] Henri Rebecq, René Ranftl, Vladlen Koltun, and Davide Scaramuzza. 2019. High Speed and High Dynamic Range Video with an Event Camera. *Computing Research Repository (CoRR)* (2019).

[38] Yaniv Romano, John Isidoro, and Peyman Milanfar. 2017. RAISR: Rapid and Accurate Image Super Resolution. *IEEE Transactions on Computational Imaging* 3, 1 (2017), 110–125.

[39] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. 2015. U-Net: Convolutional Networks for Biomedical Image Segmentation. In *Medical Image Computing and Computer-Assisted Intervention (MICCAI)*, Vol. 9351. 234–241.

[40] Cedric Scheerlinck, Nick Barnes, and Robert E. Mahony. 2018. Continuous-Time Intensity Estimation Using Event Cameras. In *Asian Conference on Computer Vision (ACCV)*, Vol. 11365. 308–324.

[41] Cedric Scheerlinck, Nick Barnes, and Robert E. Mahony. 2019. Asynchronous Spatial Image Convolutions for Event Cameras. *IEEE Robotics and Automation Letters* 4, 2 (2019), 816–822.

[42] Cedric Scheerlinck, Henri Rebecq, Daniel Gehrig, Nick Barnes, Robert E. Mahony, and Davide Scaramuzza. 2020. Fast Image Reconstruction with an Event Camera. In *IEEE Winter Conference on Applications of Computer Vision (WACV)*. 156–163.

[43] Timo Stoffregen, Cedric Scheerlinck, Davide Scaramuzza, Tom Drummond, Nick Barnes, Lindsay Kleeman, and Robert E. Mahony. 2020. Reducing the Sim-to-Real Gap for Event Cameras. In *ECCV*, Vol. 12372. 534–549.

[44] Lin Wang, S. Mohammad Mostafavi I., Yo-Sung Ho, and Kuk-Jin Yoon. 2019. Event-Based High Dynamic Range Image and Very High Frame Rate Video Generation Using Conditional Generative Adversarial Networks. In *CVPR*. 10081–10090.

[45] Chengxi Ye, Anton Mitrokhin, Chethan Parameshwara, Cornelia Fermüller, James A. Yorke, and Yiannis Aloimonos. 2018. Unsupervised Learning of Dense Optical Flow and Depth from Sparse Event Data. *Computing Research Repository (CoRR)* (2018).

[46] Linguang Zhang and Szymon Rusinkiewicz. 2018. Learning to Detect Features in Texture Images. In *CVPR*. 6325–6333.

[47] Yixin Zhang and Zilei Wang. 2020. Joint Adversarial Learning for Domain Adaptation in Semantic Segmentation. In *AAAI*. 6877–6884.

[48] Alex Zihao Zhu, Dinesh Thakur, Tolga Özaslan, Bernd Pfrommer, Vijay Kumar, and Kostas Daniilidis. 2018. The Multivehicle Stereo Event Camera Dataset: An Event Camera Dataset for 3D Perception. *IEEE Robotics and Automation Letters* 3, 3 (2018), 2032–2039.

[49] Alex Zihao Zhu, Ziyun Wang, Kaung Khant, and Kostas Daniilidis. 2019. Event-GAN: Leveraging Large Scale Image Datasets for Event Cameras. *Computing Research Repository (CoRR)* (2019).

[50] Alex Zihao Zhu, Liangzhe Yuan, Kenneth Chaney, and Kostas Daniilidis. 2018. EV-FlowNet: Self-Supervised Optical Flow Estimation for Event-based Cameras. In *Robotics: Science and Systems XIV*.

[51] Alex Zihao Zhu, Liangzhe Yuan, Kenneth Chaney, and Kostas Daniilidis. 2019. Unsupervised Event-Based Learning of Optical Flow, Depth, and Egomotion. In *CVPR*. 989–997.

[52] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A. Efros. 2017. Unpaired Image-to-Image Translation Using Cycle-Consistent Adversarial Networks. In *ICCV*. 2242–2251.

[53] Juntang Zhuang, Tommy Tang, Yifan Ding, Sekhar C. Tatikonda, Nicha C. Dvornek, Xenophon Papademetris, and James S. Duncan. 2020. AdaBelief Optimizer: Adapting Stepsizes by the Belief in Observed Gradients. In *Advances in Neural Information Processing Systems (NeurIPS)*.