Self-Supervised Learning of Event-Based Optical Flow with Spiking Neural Networks

Anonymous Author(s)
Affiliation
Address
email

Abstract

Neuromorphic sensing and computing hold a promise for highly energy-efficient and high-bandwidth-sensor processing. A major challenge for neuromorphic computing is that learning algorithms for traditional artificial neural networks (ANNs) do not transfer directly to spiking neural networks (SNNs) due to the discrete spikes and more complex neuronal dynamics. As a consequence, SNNs have not yet been successfully applied to complex, large-scale tasks. In this article, we focus on the self-supervised learning problem of optical flow estimation from event-based camera inputs, and investigate the changes that are necessary to the state-of-the-art ANN training pipeline in order to successfully tackle it with SNNs. More specifically, we first modify the input event representation to encode a much smaller time slice with minimal explicit temporal information. Consequently, we make the network's neuronal dynamics and recurrent connections responsible for integrating information over time. Moreover, we reformulate the self-supervised loss function for event-based optical flow to improve its convexity. We perform experiments with various types of recurrent ANNs and SNNs using the proposed pipeline. Concerning SNNs, we investigate the effects of elements such as parameter initialization and optimization, surrogate gradient shape, and adaptive neuronal mechanisms. We find that initialization and surrogate gradient width play a crucial part in enabling learning with sparse inputs, while the inclusion of adaptivity and learnable neuronal parameters can improve performance. We show that the performance of the proposed ANNs and SNNs are on par with that of the current state-of-the-art ANNs trained in a self-supervised manner.

1 Introduction

Neuromorphic hardware promises highly energy-efficient and low-latency sensing and processing thanks to its sparse and asynchronous nature. Event cameras capture brightness changes at microsecond resolution [16], while neuromorphic processors have demonstrated orders of magnitude lower energy consumption and latency compared to von Neumann architectures [11, 27]. One of the larger impediments to widespread neuromorphic adoption is the fact that learning algorithms designed for traditional artificial neural networks (ANNs) do not transfer one-to-one to spiking neural networks (SNNs), which exhibit sparse, binary activity and more complex neuronal dynamics. On the one hand, this has driven research into the conversion of ANNs to SNNs without loss of accuracy, but with the promised efficiency gains [39]. On the other hand, it has limited the application of directly-trained SNNs in the computer vision domain to less complicated and often discrete problems like image classification [10, 13] on constrained datasets such as N-MNIST [31] or DVS128 Gesture [2].

Still, many ongoing developments in the area of direct SNN training are promising and may form building blocks for tackling more complex tasks. Surrogate gradients [30, 38, 42, 48], which act

Submitted to 35th Conference on Neural Information Processing Systems (NeurIPS 2021). Do not distribute.

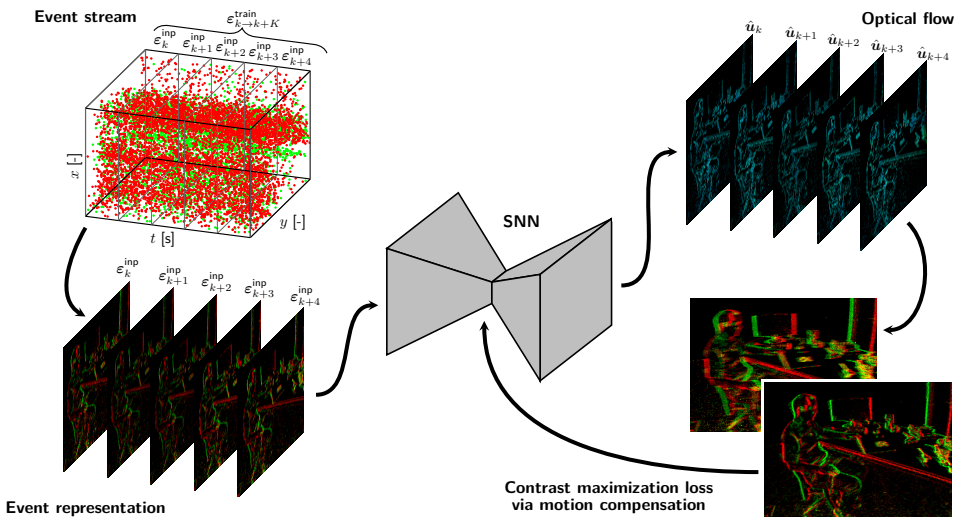

Figure 1: Self-supervised event-based optical flow pipeline for deep SNNs. In order of processing, the event stream is split into small partitions with the same number of events, which are formatted and then fed to the network in a sequential fashion. An optical flow map is predicted for each partition with which every input event is associated with a motion vector. Once a sufficient number of events has been processed, we perform a backward pass using our contrast maximization loss [15].

as stand-in for the non-differentiable spiking function in the backward pass, enable traditional backpropagation with few adjustments. Similarly, the inclusion of parameters governing the neurons' internal dynamics in the optimization was demonstrated to be beneficial [13, 35]. Many works also include some form of activity regularization to keep neurons from excessive spiking [5, 48] or to balance excitability through adaptation [4, 34]. Kickstarting initial activity (hence gradient flow) is often not the goal of these regularization terms, even though [48] shows that there is a narrow activity band in which learning is optimal. This ties in with the initialization of parameters, which has not been rigorously covered for SNNs with sparse inputs yet, leaving room for improvement.

Taking these lessons, the goal of this work is to marry the fields of neuromorphic sensing and processing. To this end, we tackle a real-world large-scale problem by learning, in a self-supervised fashion and using SNNs, to estimate the optical flow encoded in a continuous stream of events; task that is usually tackled with deep, fully convolutional ANNs [50, 52]. By focusing on such a complex problem, we aim to identify and tackle emerging knowledge gaps regarding SNN training.

In summary, the main contribution of this article is two-fold. First, we propose a novel self-supervised learning (SSL) framework for event-based optical flow estimation that puts emphasis on the networks' capacity to integrate temporal information. This training pipeline, illustrated in Fig. 1, is built around a reformulation of the self-supervised loss function from [52] that improves its convexity. Second, through this framework, we train the first set of deep SNNs that successfully solve the problem at hand. We validate our proposals through extensive quantitative and qualitative evaluations on multiple datasets. Additionally, for the SNNs, we investigate the effects of elements such as parameter initialization and optimization, surrogate gradient shape, and adaptive neuronal mechanisms.

2 Related Work

Due to the potential of event cameras to enable high-bandwidth optical flow estimation, extensive research has been conducted on this topic since these sensors were introduced [1, 6, 8, 14]. Regarding learning-based approaches, in [50], Zhu *et al.* proposed the first convolutional ANN for this task, which was trained in an SSL fashion with the supervisory signal coming from the photometric error between subsequent grayscale frames captured with the active pixel sensor (APS) of the DAVIS240C [7]. Alongside this network, the authors released the Multi-Vehicle Stereo Event Camera (MVSEC) dataset [51], the first event camera dataset with ground truth optical flow estimated from depth and ego-motion sensors. A similar SSL approach was introduced in [45], but here optical flow was

obtained through an ANN estimating depth and camera pose. Later, Zhu *et al.* refined their pipeline and, in [52], proposed an SSL framework around the contrast maximization for motion compensation idea from [14, 15]; with which, as explained in Section 3.2, the supervisory signal comes directly from the events and there is no need for additional sensors. More recently, Stoffregen and Scheerlinck *et al.* showed in [41] that, if trained with synthetic event sequences (from event camera simulators [36]) and ground truth data in a pure supervised fashion, the ANN from [50, 52] reaches higher accuracy levels when evaluated on MVSEC. Lastly, to hold up to the promise of high-speed optical flow, there has been a significant effort toward the miniaturization of optical flow ANNs [33, 26].

With respect to learning-based SNNs for optical flow estimation, only the works of Paredes-Vallés *et al.* [34] and Lee *et al.* [24, 25] are to be highlighted. In [34], the authors presented the first convolutional SNN in which motion selectivity emerges in an unsupervised fashion through Hebbian learning [21] and thanks to synaptic connections with multiple delays. However, this learning method limits the deployability of this architecture to event sequences with similar statistics to those used during training. On the other hand, in [24], the authors proposed a hybrid network integrating spiking neurons in the encoder with ANN layers in the decoder, which was trained through the SSL pipeline from [50]. This architecture was later expanded in [25] with a secondary ANN-based encoder used to retrieve information from the APS frames. Lastly, SNNs have also been implemented in neuromorphic hardware for optical flow estimation [18], although this did not involve learning. Hence, until now, no one has yet attempted the SSL of optical flow with a pure SNN approach.

Most of the SNN work in other computer vision domains has so far been focused on discrete problems like classification [10, 13, 43, 49] and binary motion-based segmentation [32]. A notable exception is the work from Gehrig *et al.* [17], who propose a convolutional spiking encoder to continuously predict angular velocities from event data. However, until now, no one has yet attempted a dense (i.e., with per-pixel estimates) regression problem with deep SNNs, requiring recurrency.

3 Method

3.1 Input event representation

An event camera consists of a pixel array that responds, in a sparse and asynchronous fashion, to changes in brightness through streams of events [16]. For an ideal camera, an event $\boldsymbol{e}_i = (\boldsymbol{x}_i, t_i, p_i)$ of polarity $p_i \in \{+, -\}$ is triggered at pixel $\boldsymbol{x}_i = (x_i, y_i)^T$ and time t_i whenever the brightness change since the last event at that pixel reaches the contrast sensitivity threshold for that polarity.

The great majority of learning-based models proposed to date for the problem of event-based optical flow estimation encode, in one form or another, spatiotemporal information into the input event representation before passing it to the neural architectures. This allows stateless (i.e., non-recurrent) ANNs to accurately estimate optical flow at the cost of having to accumulate events over relatively long time windows for their apparent motion to be perceivable. The most commonly used representations make use of multiple discretized frames of event counts [24, 25, 33, 41, 52] and/or the per-pixel average or the most recent event timestamps [26, 45, 50].

Ideally, SNNs would immediately receive spikes at event locations, which implies that temporal information should not be encoded in the input representation, but should be extracted by the network. To enforce this, we use a representation consisting only of per-pixel and per-polarity event counts, as in Fig. 1. This representation gets populated with consecutive, non-overlapping partitions of the event stream $\boldsymbol{\varepsilon}_k^{\text{inp}} \doteq \{\boldsymbol{e}_i\}_{i=0}^{N-1}$ (referred to as input partition) each containing a fixed number of events, N.

3.2 Self-supervised learning of optical flow via contrast maximization

We use the contrast maximization proxy loss for motion compensation [15] to learn to estimate optical flow from the continuous event stream in a self-supervised fashion. The idea behind this optimization framework is that accurate optical flow information is encoded in the spatiotemporal misalignments among the events triggered by the same portion of a moving edge (i.e., blur) and that, to retrieve it, one has to compensate for this motion (i.e., deblur the event partition). Knowing the per-pixel optical flow $\boldsymbol{u}(\boldsymbol{x}) = (u(\boldsymbol{x}), v(\boldsymbol{x}))^T$, the events can be propagated to a reference time t_{ref} through:

$$\boldsymbol{x}_i' = \boldsymbol{x}_i + (t_{\text{ref}} - t_i)\boldsymbol{u}(\boldsymbol{x}_i) \tag{1}$$

In this work, we reformulate the deblurring quality measure proposed by Mitrokhin *et al.* [28] and Zhu *et al.* [52]: the per-pixel and per-polarity average timestamp of the image of warped events (IWE). The lower this metric, the better the event deblurring and the more accurate the optical flow estimation. We generate an image of the average timestamp at each pixel for each polarity p' via bilinear interpolation:

$$\begin{aligned} T_{p'}(\boldsymbol{x};\boldsymbol{u}|t_{\text{ref}}) &= \frac{\sum_j \kappa(x-x'_j)\kappa(y-y'_j)t_j}{\sum_j \kappa(x-x'_j)\kappa(y-y'_j)+\epsilon} \\ \kappa(a) &= \max(0, 1-|a|) \\ j = \{i \mid p_i = p'\}, &\quad p' \in \{+,-\}, \quad \epsilon \approx 0 \end{aligned} \tag{2}$$

Previous works minimize the sum of the squared temporal images resulting from the warping process [52, 33]. In contrast, we scale this sum prior to the minimization with the number of pixels with at least one warped event for the loss function to be convex:

$$\mathcal{L}_{\text{contrast}}(t_{\text{ref}}) = \frac{\sum_{\boldsymbol{x}} T_+(\boldsymbol{x};\boldsymbol{u}|t_{\text{ref}})^2 + T_-(\boldsymbol{x};\boldsymbol{u}|t_{\text{ref}})^2}{\sum_{\boldsymbol{x}} [n(\boldsymbol{x}') > 0] + \epsilon} \tag{3}$$

where $n(\boldsymbol{x}')$ denotes a per-pixel event count of the IWE. As shown in Appendix A, without the scaling, the loss function is not well-defined as the optimal solution is to always warp events with large timestamps out of the image space so they do not contribute to Eq. 2. Previous works circumvented this issue by limiting the maximum magnitude of the optical flow vectors that could be estimated through scaled TanH activations in the prediction layers [33, 52].

As in [52], we perform the warping process both in a forward ($t_{\text{ref}}^{\text{fw}}$) and in a backward fashion ($t_{\text{ref}}^{\text{bw}}$) to prevent temporal scaling issues during backpropagation. The total loss used to train our event-based optical flow networks is then given by:

$$\mathcal{L}_{\text{contrast}} = \mathcal{L}_{\text{contrast}}(t_{\text{ref}}^{\text{fw}}) + \mathcal{L}_{\text{contrast}}(t_{\text{ref}}^{\text{bw}}) \tag{4}$$

$$\mathcal{L}_{\text{flow}} = \mathcal{L}_{\text{contrast}} + \lambda \mathcal{L}_{\text{smooth}} \tag{5}$$

where λ is a scalar balancing the effect of the two losses and $\mathcal{L}_{\text{smooth}}$ is a Charbonnier smoothness prior [9], as in [50, 52]. Since $\mathcal{L}_{\text{contrast}}$ does not propagate the error back to pixels without input events, we mask the output of our networks so that null optical flow vectors are returned at these pixels locations. Furthermore, we mask the computation of $\mathcal{L}_{\text{smooth}}$ so that this regularization mechanism only considers optical flow estimates from neighboring pixels with at least one event.

As hinted by the motion model in Eq. 1 and discussed in [15, 40], there has to be enough linear blur in the input event partition for $\mathcal{L}_{\text{contrast}}$ to be a robust supervisory signal. This is usually not the case in our training pipeline due to the small number of input events N that we pass to the networks at each forward pass. For this reason, we define a secondary event partition, the so-called training partition $\varepsilon_{k\to k+K}^{\text{train}} \doteq \{(\varepsilon_i^{\text{inp}}, \hat{\boldsymbol{u}}_i)\}_{i=k}^{K}$, which is a buffer that gets populated every forward pass with an input event partition and its corresponding optical flow estimates. At training time, we perform a backward pass with the content of the buffer using backpropagation through time once it contains K successive event-flow tuples; after which we detach the state of the networks from the computational graph and clear the buffer. Note that $\mathcal{L}_{\text{smooth}}$ is also applied in the temporal dimension by smoothing optical flow estimates at the same pixel location from adjacent tuples.

3.3 Spiking neuron models

In this work, we compare various spiking neuron models from literature on the task of event-based optical flow estimation. All models are based on the leaky-integrate-and-fire (LIF) neuron, whose membrane potential U and synaptic input current I at timestep k can be written as:

$$U_i^k = (1 - S_i^{k-1})\alpha U_i^{k-1} + (1-\alpha) I_i^k \tag{6}$$

$$I_i^k = \sum_j W_{ij}^{\text{ff}} S_j^k + \sum_r W_{ir}^{\text{rec}} S_r^{k-1} \tag{7}$$

where j and r denote presynaptic neurons while i is for postsynaptic, α is the membrane decay or leak, $S \in \{0, 1\}$ a neuron spike, and W^{ff} and W^{rec} feedforward and recurrent connections, respectively.

Membrane decays can either be fixed or learned. A neuron fires an output spike S if the membrane potential exceeds a threshold θ, which can either be fixed, learned, or adaptive (see below). Firing also triggers a reset of U, which is either *hard*, as in Eq. 6, or *soft*, as in [4]. However, the former is said to be more suited for deeper networks, as it gets rid of errors accumulated by the surrogate gradient [23].

Following [4], we introduce an adaptive threshold to make up the adaptive-LIF (ALIF) model. A second state variable T acts as a low-pass filter over the output spikes, adapting the firing threshold based on the neuron's activity:

$$\theta_i^k = \beta_0 + \beta_1 T_i^k \tag{8}$$

$$T_i^k = \eta T_i^{k-1} + (1-\eta) S_i^{k-1} \tag{9}$$

where $\beta_{\{0,1\}}$ are (learnable) constants, and η is the (learnable) threshold decay/leak. ALIF's equation for U and I are identical to the LIF formulation. By decaying the threshold very slowly, T can act as a longer-term memory of the neuron [5].

Instead of postsynaptic adaptivity, we can keep a trace of presynaptic activity and use that to regularize neuron firing, giving the presynaptic-LIF (PLIF) model. The authors of [34] implement this kind of adaptation mechanism by subtracting a presynaptic trace P from the synaptic input current:

$$I_i^k = \sum_j W_{ij}^{\text{ff}} S_j^k + \sum_r W_{ir}^{\text{rec}} S_r^{k-1} - \rho_0 P_i^k \tag{10}$$

$$P_i^k = \rho_1 P_i^{k-1} + \frac{1-\rho_1}{|R_i|} \sum_{j \in R_i} S_j^{k-1} \tag{11}$$

where $\rho_{\{0,1\}}$ are (learnable) addition and decay constants, and R_i is the set of receptive fields of neuron i over all channels (i.e., the second term in Eq. 11 is an average pooling averaged over all channels). Adaptation based on presynaptic instead of postsynaptic activity minimizes adaptation delay, making it especially suited to the fast-changing nature of event data [34]. In this spirit, we also propose the XLIF model, a crossover between ALIF and PLIF, which adapts its threshold based on presynaptic activity:

$$\theta_i^k = \beta_0 + \beta_1 P_i^k \tag{12}$$

As surrogate gradient for the spiking function, due to its computational tractability, we opt for the derivative of the inverse tangent $\sigma'(x) = 1/(1+\gamma x^2)$ as in [13], with γ being the surrogate width and $x = U - \theta$. In order to ensure gradient flow (hence learning) in the absence of neuron firing, the width should be sufficient to cover at least a range of subthreshold membrane potentials, while the height should be properly scaled (i.e., ≤ 1) for stable learning [48]. Exact shape is of less importance for final accuracy. Further details on all hyperparameters can be found in Appendix G.

3.4 Network architectures

We evaluate the two trends on neural network design for event cameras through (spiking) recurrent variants of EV-FlowNet [50] (encoder-decoder) and FireNet [37] (lightweight, no downsampling). An overview of the evaluated architectures can be found in Fig. 2. The use of explicit recurrent connections in all our ANNs and SNNs is justified through the ablation study in Appendix D.

On the one hand, the base architecture referred to as EV-FlowNet is a recurrent version of the network proposed in [50]. Once encoded as in Section 3.1, the input event partition is passed through four recurrent encoders performing strided convolution followed by ConvGRU [3] with output channels doubling after each encoder (starting from 32), two residual blocks [19], and four decoder layers that perform bilinear upsampling followed by convolution. After each decoder, there is a (concatenated) skip connection from the corresponding encoder, as well as a depthwise (i.e., 1×1) convolution to produce a lower scale flow estimate, which is then concatenated with the activations of the previous decoder. The $\mathcal{L}_{\text{flow}}$ loss (see Eq. 5) is applied to each intermediate optical flow estimate via upsampling. All layers use 3×3 kernels and ReLU activations except for the prediction layers, which use TanH activations.

On the other hand, the FireNet architecture in Fig. 2 is an adaptation of the lightweight network proposed in [37], which was originally designed for event-based image reconstruction. However, as

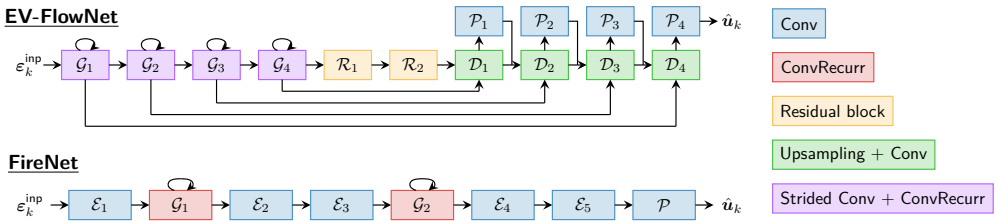

Figure 2: Schematic of the base neural networks used in this work. All evaluated variants inherit from these architectures and only vary the neuron model and/or the convolutional recurrent layer.

shown in [33], this architecture is also suitable for fast optical flow estimation. The base architecture consists of five encoder layers that perform single-strided convolutions, two ConvGRUs, and a final prediction layer that performs depthwise convolution. All layers have 32 output channels and use 3×3 kernels and ReLU activations except for the final layer, which uses a TanH activation.

Based on these architectures, we have designed several variants: (i) RNN-EV-FlowNet and RNN-FireNet, which use vanilla ConvRNNs (see Appendix B) instead of ConvGRUs; (ii) Leaky-EV-FlowNet and Leaky-FireNet, which use ConvRNNs and whose neurons are stateful cells with leaks (for a more direct comparison with the SNNs, see Appendix B); and (iii) SNN-EV-FlowNet and SNN-FireNet (with SNN being LIF, ALIF, PLIF or XLIF), the SNN variants that use ConvRNNs and whose neurons are spiking and stateful according to the neuron models in Section 3.3.

The prediction layers of all SNN variants are kept real-valued with TanH activation, acting as a learned decoder from binary spikes to a dense optical flow estimate. The first layer of the SNNs can likewise be viewed as a learned spike encoder, receiving integer event counts and emitting spikes.

4 Experiments

To highlight the robustness of our SSL pipeline, we train our networks on the indoor forward-facing sequences from the UZH-FPV Drone Racing Dataset [12], which is characterized by a much wider distribution of optical flow vectors than the datasets that we use for evaluation, i.e., MVSEC [50], High Quality Frames (HQF) [41], and the Event-Camera Dataset (ECD) [29]. The selected training sequences consist of approximately 15 minutes of event data that we split into 140 128×128 (randomly cropped) sequences with 500k events each. We further augment this data using random horizontal, vertical, and polarity flips.

Our framework is implemented in PyTorch[1]. We use the Adam optimizer [22] and a learning rate of 0.0002, and train with a batch size of 8 for 100 epochs. We fix the number of events for each input partition to $N = 1$k, while we use 10k events for each training event partition. This is equivalent to $K = 10$ forward passes per backward pass (i.e., the network's unrolling), as described in Section 3.2 and illustrated in Fig. 1. Lastly, we empirically set the scaling weight for $\mathcal{L}_{\text{smooth}}$ to $\lambda = 0.001$.

We evaluated our architectures on the MVSEC dataset [51] with the ground-truth optical flow data provided by Zhu *et al.* in [50], which was generated at each APS frame timestamp, and scaled to be the displacement for the duration of one ($\text{dt} = 1$) and four APS frames ($\text{dt} = 4$). Optical flow predictions were also generated at each frame timestamp by using all the events in the time window as input for $\text{dt} = 1$, while only 25% of the window events at a time for $\text{dt} = 4$ (due to the larger displacements). For comparison against the ground truth, the predicted optical flow is converted from units of pixels/partition to units of pixel displacement by multiplying it with $\text{dt}_{\text{gt}}/\text{dt}_{\text{input}}$. We compare our recurrent ANNs and SNNs against the state-of-the-art on self-supervised event-based optical flow estimation: the original (non-recurrent) EV-FlowNet [50] trained with either photometric error as in [50] or contrast maximization [52], and the hybrid SNN-ANN network from [24]. Quantitative results of this evaluation are presented in Table 1. We report the average endpoint error (AEE) and the percentage of points with AEE greater than 3 pixels and 5% of the magnitude of the optical flow vector, denoted by $\%_{\text{Outlier}}$, over pixels with valid ground truth data and at least one input event. Qualitative results of our best performing networks on this dataset are shown in Fig. 3.

[1]Our code will be made open-source upon publication.

Table 1: Quantitative evaluation on MVSEC [51]. For each sequence, we report the AEE (lower is better, ↓) in pixels and the percentage of outliers, $\%_{\text{Outlier}}$ (↓). Best in bold, runner up underlined.

dt = 1	outdoor_day1		indoor_flying1		indoor_flying2		indoor_flying3	
	AEE	$\%_{\text{Outlier}}$	AEE	$\%_{\text{Outlier}}$	AEE	$\%_{\text{Outlier}}$	AEE	$\%_{\text{Outlier}}$
EV-FlowNet* [50]	0.49	0.20	1.03	2.20	1.72	15.10	1.53	11.90
EV-FlowNet* [52]	**0.32**	**0.00**	**0.58**	**0.00**	**1.02**	**4.00**	**0.87**	**3.00**
Hybrid-EV-FlowNet* [24]	0.49	-	0.84	-	1.28	-	1.11	-
EV-FlowNet	0.47	0.25	0.60	0.51	1.17	8.06	0.93	5.64
RNN-EV-FlowNet	0.56	1.09	0.62	0.97	1.20	8.82	0.93	5.51
Leaky-EV-FlowNet	0.53	0.28	0.71	0.60	1.43	11.37	1.14	8.12
LIF-EV-FlowNet	0.53	0.33	0.71	1.41	1.44	12.75	1.16	9.11
ALIF-EV-FlowNet	0.57	0.42	1.00	2.46	1.78	17.69	1.55	15.24
PLIF-EV-FlowNet	0.60	0.52	0.75	0.85	1.52	13.38	1.23	9.48
XLIF-EV-FlowNet	0.45	0.16	0.73	0.92	1.45	12.18	1.17	8.35
FireNet	0.55	0.35	0.89	1.93	1.62	14.65	1.35	10.64
RNN-FireNet	0.62	0.52	0.96	2.60	1.77	17.55	1.48	13.60
Leaky-FireNet	0.52	0.41	0.90	2.66	1.67	16.09	1.43	13.16
LIF-FireNet	0.57	0.40	0.98	2.48	1.77	16.40	1.50	12.81
ALIF-FireNet	0.62	0.45	1.04	3.02	1.85	18.88	1.58	15.00
PLIF-FireNet	0.56	0.38	0.90	1.93	1.67	14.47	1.41	11.17
XLIF-FireNet	0.54	0.34	0.98	2.75	1.82	18.19	1.54	14.57
dt = 4								
EV-FlowNet* [50]	1.23	**7.30**	2.25	24.70	4.05	45.30	3.45	39.70
EV-FlowNet* [52]	1.30	9.70	2.18	24.20	3.85	46.80	3.18	47.80
Hybrid-EV-FlowNet* [24]	**1.09**	-	2.24	-	**3.83**	-	3.18	-
EV-FlowNet	1.69	12.50	**2.16**	**21.51**	3.90	**40.72**	**3.00**	**29.60**
RNN-EV-FlowNet	1.91	16.39	2.23	22.10	4.01	41.74	3.07	30.87
Leaky-EV-FlowNet	1.99	17.86	2.59	30.71	4.94	54.74	3.84	42.33
LIF-EV-FlowNet	2.02	18.91	2.63	29.55	4.93	51.10	3.88	41.49
ALIF-EV-FlowNet	2.13	20.96	3.81	50.36	6.40	66.03	5.53	61.07
PLIF-EV-FlowNet	2.24	23.76	2.80	34.34	5.21	52.98	4.12	45.31
XLIF-EV-FlowNet	1.67	12.69	2.72	31.69	4.93	51.36	3.91	42.52
FireNet	2.04	20.93	3.35	42.50	5.71	61.03	4.68	53.42
RNN-FireNet	2.35	24.31	3.64	46.54	6.33	63.89	5.20	56.60
Leaky-FireNet	1.96	18.26	3.42	42.03	5.92	58.80	4.98	52.57
LIF-FireNet	2.12	21.00	3.72	48.27	6.27	64.16	5.23	58.43
ALIF-FireNet	2.36	25.82	3.94	52.35	6.65	67.61	5.60	61.93
PLIF-FireNet	2.11	20.64	3.44	44.02	5.94	64.02	4.98	57.53
XLIF-FireNet	2.07	18.83	3.73	47.89	6.51	67.25	5.43	60.59

*Non-recurrent ANNs with input event representations encoding spatiotemporal information, as described in [24, 50, 52].

For the sake of completeness, as in [41, 33] we also evaluate our architectures on the ECD [29] and HQF [41] datasets. The details and results of this evaluation can be found in Appendix C. Due to the lack of ground truth data in these datasets, we assess the quality of the estimated optical flow based on metrics derived from the contrast maximization framework [14, 15].

Regarding the SNN variants, the results in Table 1, Fig. 3 and Appendix C correspond to networks whose neuronal parameters (i.e., leaks, thresholds, adaptive mechanisms) were also optimized when applicable. See Appendix E for an ablation study on the learnable parameters; further details regarding parameter settings and initialization are given in Appendix G.

4.1 Evaluation of the ANN and SNN architectures

Firstly, quantitative results in Table 1 confirms the validity of the proposed SSL framework for event-based optical flow estimation with recurrent networks. As shown, our base architectures EV-FlowNet and FireNet perform on par with the current state-of-the-art, even though these non-recurrent networks from literature encode explicit temporal information in their input event representations, and were trained on other very similar sequences from MVSEC [51] to prevent deviations of the

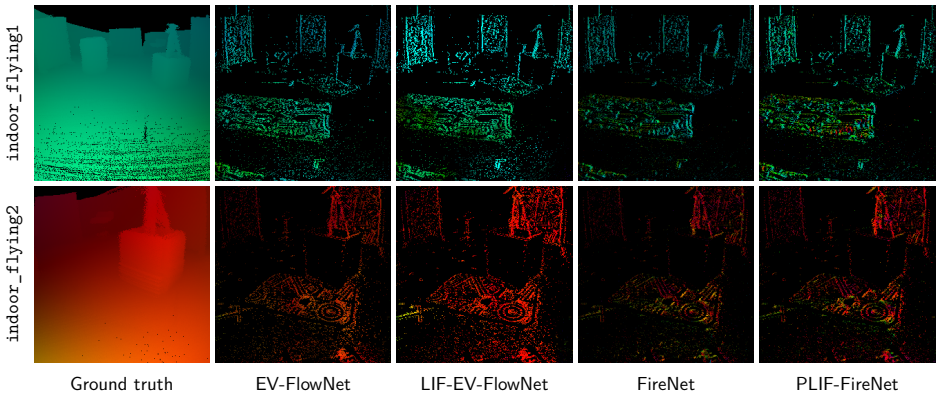

Figure 3: Qualitative evaluation of our best performing ANNs and SNNs on sequences from the MVSEC dataset [51]. The optical flow color-coding scheme and can be found in Appendix C.

input statistics from the training distribution during inference [24, 50, 52]. Since we train on a very different dataset [12], this on-par performance thus also confirm the generalizability of our ANNs and SNNs to distinctly different scenes and distributions of optical flow vectors. This claim is further supported by qualitative results in Fig. 3 and additional results in Appendix C.

Secondly, from the comparison between our base ANN architectures and their spiking counterparts without adaptation mechanisms (i.e., LIF-EV-FlowNet and LIF-FireNet), we can conclude that, although there is a general increase in the AEE and the percentage of outliers when going spiking, the proposed SNNs are still able to produce high quality event-based optical flow estimates. In fact, according to Table 1, the main drop in accuracy does not come from the incorporation of the spiking function (and the selection of the derivative of the inverse tangent as surrogate gradient [13]), but mainly from the use of vanilla convolutional recurrent layers instead of gated recurrent units. As shown, our spiking LIF architectures perform very close to their RNN and leaky counterparts, despite the latter being ANNs. This highlights the important need for more powerful convolutional recurrent units for SNNs, similar to ConvLSTMs [44] and ConvGRUs [3] for ANNs, as this would narrow the performance gap between these two processing modalities according to our observations. Interestingly, a previous comparison of the performance of recurrent ANNs and SNNs for event-based classification [20] suggested similar improvements to SNN units.

4.2 Impact of adaptive mechanisms for spiking neurons

Table 1 and Appendix C also allow us to draw conclusions about the effectiveness of the adaptive mechanisms for spiking neurons introduced in Section 3.3. For both EV-FlowNet and FireNet, we observe that threshold adaptation based on postsynaptic activity (i.e., the ALIF model) performs worse compared to other models. The loss curves in Appendix F support this observation. While the ALIF model was shown to be effective for learning long temporal dependencies from relatively low-dimensional data as in [4, 5, 46], the adaptation delay introduced by relying on a postsynaptic signal seems detrimental when working with fast-changing, high-dimensional event data. This is in line with suggestions by Paredes-Vallés *et al.* in [34], who use presynaptic adaptation for this reason. Our own results with presynaptic adaptation (i.e., PLIF and XLIF models) are somewhat inconclusive. While PLIF performs better in the case of FireNet, this is not the case for EV-FlowNet. On the other hand, XLIF's performance is very similar to the LIF model for both FireNet and EV-FlowNet architectures. Based on these observations, we conclude that adaptivity based on presynaptic activity should be considered for further development. In this regard, the XLIF model has the advantage that it is able to generate activity (leading to gradient flow, and thus learning) even for very small inputs, whereas PLIF is incapable of this (P is always positive) for a given threshold.

4.3 Further lessons on training deep SNNs

Multiple problems arise when training deep SNNs for a regression task that involves sparse data, as is done here. Regarding learning, we find that gradient vanishing poses the main issue. Even considering

dense inputs/loss and a shallow (in timesteps or in layers) SNN, sufficient gradient flow is a result of wide enough (covering at least some range of subthreshold potentials) and properly scaled (i.e., ≤ 1) surrogate gradients [23, 46, 48], and parameter initializations that lead to non-negligible amounts of spiking activity [48]. Sparse data and deep networks make finding the proper settings more difficult, and for this reason, we have tried to increase the robustness of various of these hyperparameter settings. First, we looked at the learning performance and gradient flow of networks with various surrogate gradient shapes and widths. Compared to the surrogate specified in Section 3.3, SuperSpike [47] with $\gamma = 10$ and $\gamma = 100$ (both narrower) show little learning due to negligible gradient flow (see Appendix H for more details). One way of reducing the effect of a too narrow surrogate gradient would be to trigger spiking activity through regularization terms in the loss function, as done in, e.g., [5, 48]. These form a direct connection between loss and the neuron in question, bypassing most of the gradient vanishing that would happen in later layers. We tried the variant proposed in [48], which is aimed at achieving at least a certain fraction of neurons to be active at any given time. With this fraction set to 5%, we saw that for SuperSpike with $\gamma = 10$ there was some learning happening, while for $\gamma = 100$ there was no effect. Plots of the loss curves and gradient magnitudes are available in Appendix H. Of course, more research into these and other regularization methods is necessary. Alternatively, as done in [23], batch normalization (or other presynaptic normalization mechanisms) could be used to ensure proper activity and gradient flow.

Regarding the network output, there seems to be an intuitive gap between classification and regression tasks, with the latter requiring a higher resolution to be solved successfully. In our view, there are two aspects to this that might pose an issue to SNNs. First, given the single prediction layer that the here-presented SNNs have to go from binary to real-valued activations, one could expect a loss in output resolution compared to equivalent ANN architectures. Second, even for moderate activity levels, the outputs of a spiking layer can be much larger in magnitude than an equivalent ANN layer, even for comparable parameter initializations. Intuitive solutions to these shortcomings are (i) to increase the number of channels to increase the resolution, and (ii) to initialize the weights of the non-spiking output layer as to have a smaller magnitude. While increasing the number of channels did not lead to significantly improved performance or learning speed (possibly due to the fact that the weight initialization also depends on the number of channels, leading to lower activations at the same time), decreasing the initialization magnitude of the weights in the output layer did. As the loss curves in Appendix H show, the improved initialization leads to faster convergence and less variability across neuron models and the selection of learnable parameters.

5 Conclusion

In this article, we presented the first set of deep SNNs to successfully solve the real-world large-scale problem of event-based optical flow estimation. To achieve this, we first reformulated the state-of-the-art training pipeline for ANNs to shorten the time windows presented to the networks, approximating the way in which SNNs would receive spikes directly from the event camera. Additionally, we also reformulated the state-of-the-art self-supervised loss function to improve its convexity. Prior to their training with this framework, we augmented several ANN architectures from literature with explicit and/or implicit recurrency, besides the addition of the spiking behavior. Extensive quantitative and qualitative evaluations were conducted on multiple datasets. Results confirm not only the validity of our training pipeline, but also the on-par performance of the proposed set of recurrent ANNs and SNNs with the self-supervised state-of-the-art. To the best of our knowledge, and especially due to the addition of explicit recurrent connections, the proposed SNNs correspond to the most complex spiking networks in the computer vision literature, architecturally speaking. For the SNNs, we also conducted several additional studies and (i) concluded that parameter initialization and the width of the surrogate gradient have a significant impact on learning: smaller weights in the output layer speed up convergence, while a too narrow surrogate gradient prevents learning altogether; and (ii) we observed that adaptive mechanisms based on presynaptic activity outperform those based on postsynaptic activity, and perform similarly or better than the baseline without adaptation. Overall, we believe this article sets the groundwork for future research on neuromorphic processing for not only the event-based structure-from-motion problem, but also for other, similarly complex computer vision applications. For example, our results suggest the need for more powerful recurrent units for SNNs. On another note, future work should also focus on the implementation of these deep SNNs on neuromorphic hardware, as it is there where these architectures excel due to the power efficiency that their sparse and asynchronous nature entails.

References

[1] Mohammed Almatrafi, Raymond Baldwin, Kiyoharu Aizawa, and Keigo Hirakawa. Distance surface for event-based optical flow. *IEEE Trans. Pattern Anal. and Mach. Intell.*, 42(7):1547–1556, 2020.

[2] Arnon Amir, Brian Taba, David Berg, Timothy Melano, Jeffrey McKinstry, Carmelo Di Nolfo, Tapan Nayak, Alexander Andreopoulos, Guillaume Garreau, Marcela Mendoza, et al. A low power, fully event-based gesture recognition system. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 7243–7252, 2017.

[3] Nicolas Ballas, Li Yao, Chris Pal, and Aaron Courville. Delving deeper into convolutional networks for learning video representations. *Int. Conf. Learn. Representations*, 2015.

[4] Guillaume Bellec, Darjan Salaj, Anand Subramoney, Robert Legenstein, and Wolfgang Maass. Long short-term memory and learning-to-learn in networks of spiking neurons. In *Advances in Neural Information Process. Syst.*, volume 31, pages 787–797, 2018.

[5] Guillaume Bellec, Franz Scherr, Anand Subramoney, Elias Hajek, Darjan Salaj, Robert Legenstein, and Wolfgang Maass. A solution to the learning dilemma for recurrent networks of spiking neurons. *Nature Commun.*, 11(1):3625, 2020.

[6] Ryad Benosman, Charles Clercq, Xavier Lagorce, Sio-Hoi Ieng, and Chiara Bartolozzi. Event-based visual flow. *IEEE Trans. Neural Netw. Learn. Syst.*, 25(2):407–417, 2013.

[7] Christian Brandli, Raphael Berner, Minhao Yang, Shih-Chii Liu, and Tobi Delbruck. A 240$\times$180 130 dB 3μs latency global shutter spatiotemporal vision sensor. *IEEE J. Solid-State Circuits*, 49(10):2333–2341, 2014.

[8] Tobias Brosch, Stephan Tschechne, and Heiko Neumann. On event-based optical flow detection. *Frontiers in Neuroscience*, 9:137, 2015.

[9] Pierre Charbonnier, Laure Blanc-Feraud, Gilles Aubert, and Michel Barlaud. Two deterministic half-quadratic regularization algorithms for computed imaging. In *IEEE Int. Conf. Image Process.*, volume 2, pages 168–172, 1994.

[10] Loïc Cordone, Benoît Miramond, and Sonia Ferrante. Learning from event cameras with sparse spiking convolutional neural networks. *arXiv:2104.12579*, 2021.

[11] Mike Davies, Andreas Wild, Garrick Orchard, Yulia Sandamirskaya, Gabriel A. Fonseca Guerra, Prasad Joshi, Philipp Plank, and Sumedh R. Risbud. Advancing neuromorphic computing with Loihi: A survey of results and outlook. *IEEE Proc.*, pages 1–24, 2021.

[12] Jeffrey Delmerico, Titus Cieslewski, Henri Rebecq, Matthias Faessler, and Davide Scaramuzza. Are we ready for autonomous drone racing? The UZH-FPV drone racing dataset. In *IEEE Int. Conf. Robot. Autom.*, pages 6713–6719, 2019.

[13] Wei Fang, Zhaofei Yu, Yanqi Chen, Timothee Masquelier, Tiejun Huang, and Yonghong Tian. Incorporating learnable membrane time constant to enhance learning of spiking neural networks. *arXiv:2007.05785*, 2020.

[14] Guillermo Gallego, Henri Rebecq, and Davide Scaramuzza. A unifying contrast maximization framework for event cameras, with applications to motion, depth, and optical flow estimation. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 3867–3876, 2018.

[15] Guillermo Gallego, Mathias Gehrig, and Davide Scaramuzza. Focus is all you need: Loss functions for event-based vis. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 12280–12289, 2019.

[16] Guillermo Gallego, Tobi Delbruck, Garrick Orchard, Chiara Bartolozzi, Brian Taba, Andrea Censi, Stefan Leutenegger, Andrew Davison, Jörg Conradt, Kostas Daniilidis, et al. Event-based vision: A survey. *IEEE Trans. Pattern Anal. and Mach. Intell.*, 2020.

[17] M. Gehrig, S. B. Shrestha, D. Mouritzen, and D. Scaramuzza. Event-based angular velocity regression with spiking networks. In *IEEE Int. Conf. Robot. Autom.*, pages 4195–4202, 2020.

[18] Germain Haessig, Andrew Cassidy, Rodrigo Alvarez, Ryad Benosman, and Garrick Orchard. Spiking optical flow for event-based sensors using ibm's truenorth neurosynaptic system. *IEEE Trans. Biomed. Circuits Syst.*, 12(4):860–870, 2018.

[19] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 770–778, 2016.

[20] Weihua He, YuJie Wu, Lei Deng, Guoqi Li, Haoyu Wang, Yang Tian, Wei Ding, Wenhui Wang, and Yuan Xie. Comparing SNNs and RNNs on neuromorphic vision datasets: Similarities and differences. *Neural Networks*, 132:108–120, 2020.

[21] Donald O. Hebb. *The organization of behavior: A neuropsychological theory*. Wiley, 1952.

[22] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *Int. Conf. Learn. Representations*, 2014.

[23] Eimantas Ledinauskas, Julius Ruseckas, Alfonsas Juršėnas, and Giedrius Buračas. Training deep spiking neural networks. *arXiv:2006.04436*, 2020.

[24] Chankyu Lee, Adarsh Kumar Kosta, Alex Z. Zhu, Kenneth Chaney, Kostas Daniilidis, and Kaushik Roy. Spike-FlowNet: Event-based optical flow estimation with energy-efficient hybrid neural networks. In *European Conf. Comput. Vis.*, pages 366–382, 2020.

[25] Chankyu Lee, Adarsh Kumar Kosta, and Kaushik Roy. Fusion-FlowNet: Energy-efficient optical flow estimation using sensor fusion and deep fused spiking-analog network architectures. *arXiv preprint arXiv:2103.10592*, 2021.

[26] Zhuoyan Li, Jiawei Shen, and Ruitao Liu. A lightweight network to learn optical flow from event data. In *Int. Conf. on Pattern Recog.*, pages 1–7, 2021.

[27] Paul A. Merolla, John V. Arthur, Rodrigo Alvarez-Icaza, Andrew S. Cassidy, Jun Sawada, Filipp Akopyan, Bryan L. Jackson, Nabil Imam, Chen Guo, Yutaka Nakamura, et al. A million spiking-neuron integrated circuit with a scalable communication network and interface. *Science*, 345(6197):668–673, 2014.

[28] Anton Mitrokhin, Cornelia Fermüller, Chethan Parameshwara, and Yiannis Aloimonos. Event-based moving object detection and tracking. In *IEEE/RSJ Int. Conf. Intell. Robots and Syst.*, 2018.

[29] Elias Mueggler, Henri Rebecq, Guillermo Gallego, Tobi Delbruck, and Davide Scaramuzza. The event-camera dataset and simulator: Event-based data for pose estimation, visual odometry, and SLAM. *Int. J. Robot. Research*, 36(2):142–149, 2017.

[30] Emre O. Neftci, Hesham Mostafa, and Friedemann Zenke. Surrogate gradient learning in spiking neural networks: Bringing the power of gradient-based optimization to spiking neural networks. *IEEE Signal Process. Magazine*, 36(6):51–63, 2019.

[31] Garrick Orchard, Ajinkya Jayawant, Gregory K Cohen, and Nitish Thakor. Converting static image datasets to spiking neuromorphic datasets using saccades. *Frontiers in Neuroscience*, 9:437, 2015.

[32] Chethan M. Parameshwara, Simin Li, Cornelia Fermüller, Nitin J. Sanket, Matthew S. Evanusa, and Yiannis Aloimonos. SpikeMS: Deep spiking neural network for motion segmentation. *arXiv:2105.06562*, 2021.

[33] Federico Paredes-Vallés and Guido C. H. E. de Croon. Back to event basics: Self-supervised learning of image reconstruction for event cameras via photometric constancy. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2021.

[34] Federico Paredes-Vallés, Kirk Y. W. Scheper, and Guido C. H. E. De Croon. Unsupervised learning of a hierarchical spiking neural network for optical flow estimation: From events to global motion perception. *IEEE Trans. Pattern Anal. and Mach. Intell.*, 42(8):2051–2064, 2020.

[35] Nicolas Perez-Nieves, Vincent C. H. Leung, Pier Luigi Dragotti, and Dan F. M. Goodman. Neural heterogeneity promotes robust learning. *bioRxiv*, page 2020.12.18.423468, 2021.

[36] Henri Rebecq, Daniel Gehrig, and Davide Scaramuzza. ESIM: An open event camera simulator. In *Conf. Robot Learn.*, pages 969–982, 2018.

[37] Cedric Scheerlinck, Henri Rebecq, Daniel Gehrig, Nick Barnes, Robert Mahony, and Davide Scaramuzza. Fast image reconstruction with an event camera. In *IEEE Winter Conf. Appl. Comput. Vis. (WACV)*, pages 156–163, 2020.

[38] Sumit B. Shrestha and Garrick Orchard. Slayer: Spike layer error reassignment in time. In *Advances in Neural Information Process. Syst.*, 2018.

[39] Christoph Stöckl and Wolfgang Maass. Optimized spiking neurons can classify images with high accuracy through temporal coding with two spikes. *Nature Mach. Intell.*, pages 1–9, 2021.

[40] Timo Stoffregen and Lindsay Kleeman. Event cameras, contrast maximization and reward functions: An analysis. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 12300–12308, 2019.

[41] Timo Stoffregen, Cedric Scheerlinck, Davide Scaramuzza, Tom Drummond, Nick Barnes, Lindsay Kleeman, and Robert Mahony. Reducing the sim-to-real gap for event cameras. In *European Conf. Comput. Vis.*, 2020.

[42] Yujie Wu, Lei Deng, Guoqi Li, Jun Zhu, and Luping Shi. Spatio-temporal backpropagation for training high-performance spiking neural networks. *Frontiers in Neuroscience*, 12, 2018.

[43] Yannan Xing, Gaetano Di Caterina, and John Soraghan. A new spiking convolutional recurrent neural network (SCRNN) with applications to event-based hand gesture recognition. *Frontiers in Neuroscience*, 14, 2020.

[44] Shi Xingjian, Zhourong Chen, Hao Wang, Dit-Yan Yeung, Wai-Kin Wong, and Wang-Chun Woo. Convolutional LSTM network: A machine learning approach for precipitation nowcasting. In *Advances in Neural Information Process. Syst.*, pages 802–810, 2015.

[45] Chengxi Ye, Anton Mitrokhin, Cornelia Fermüller, James A. Yorke, and Yiannis Aloimonos. Unsupervised learning of dense optical flow, depth and egomotion from sparse event data. *arXiv:1809.08625*, 2018.

[46] Bojian Yin, Federico Corradi, and Sander M. Bohte. Accurate and efficient time-domain classification with adaptive spiking recurrent neural networks. *arXiv:2103.12593*, 2021.

[47] Friedemann Zenke and Surya Ganguli. SuperSpike: Supervised learning in multilayer spiking neural networks. *Neural Computation*, 30(6):1514–1541, 2018.

[48] Friedemann Zenke and Tim P. Vogels. The remarkable robustness of surrogate gradient learning for instilling complex function in spiking neural networks. *Neural Computation*, pages 1–27, 2021.

[49] Hanle Zheng, Yujie Wu, Lei Deng, Yifan Hu, and Guoqi Li. Going deeper with directly-trained larger spiking neural networks. *arXiv:2011.05280*, 2020.

[50] Alex Z. Zhu and Liangzhe Yuan. EV-FlowNet: Self-supervised optical flow estimation for event-based cameras. In *Robot.: Science and Syst.*, 2018.

[51] Alex Z. Zhu, Dinesh Thakur, Tolga Özaslan, Bernd Pfrommer, Vijay Kumar, and Kostas Daniilidis. The multivehicle stereo event camera dataset: An event camera dataset for 3D perception. *IEEE Robot. and Autom. Lett. (RA-L)*, 3(3):2032–2039, 2018.

[52] Alex Z. Zhu, Liangzhe Yuan, Kenneth Chaney, and Kostas Daniilidis. Unsupervised event-based learning of optical flow, depth, and egomotion. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 989–997, 2019.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes] The methodology behind the main contributions as stated in the abstract and introduction can be found in Section 3, while the results supporting these claims can be found in Section 4 and in the supplementary material.
   (b) Did you describe the limitations of your work? [Yes] The experiments conducted in Section 4 were designed to highlight the main limitations of the proposed SNNs with respect to traditional ANNs. For example, in Section 4.1, we describe that there is a performance gap between them, especially due to the lack of powerful gated recurrent units for SNNs.
   (c) Did you discuss any potential negative societal impacts of your work? [No] Our work does not have any potential negative societal impact that we can foresee according to the ethics guidelines.
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [N/A]
   (b) Did you include complete proofs of all theoretical results? [N/A]
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] The code needed to train and/or evaluate the proposed architectures is submitted together with the supplementary material.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] All the details are specified in Sections 3 and 4, and in the supplementary material.
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [No] Due to the amount of models trained, there was insufficient time to do multiple runs for each of them. We do however believe that, due to the many architecture variants and different experiments, our results are representative.
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [No] We trained our architectures on our internal cluster consisting of four NVIDIA GeForce GTX 1080 Ti GPUs. Each FireNet variant required approximately 14 hours to train on a single GPU, while EV-FlowNet variants required around 18 hours. We did not specify this information in our work because the computational efficiency of our code/networks was not discussed.
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes] Four (already existing) datasets were used in this work. References to their scientific articles can be found throughout the paper, but more easily in Section 4.
   (b) Did you mention the license of the assets? [No] However, all the existing datasets used in this work were previously released under a Creative Commons license or are open-source without a license.
   (c) Did you include any new assets either in the supplemental material or as a URL? [No] No new assets are presented alongside this work.
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [No] All the existing datasets we used in this work are publicly available (both scientific paper and data), so there was no need to ask the authors for consent.
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [No] However, all the existing datasets we used in this work were previously published on peer-reviewed conferences or journals such as RA-L, ICRA and ECCV.

5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]