Neuromorphic Optical Flow and Real-time Implementation with Event Cameras

Yannick Schnider\textsuperscript{1,2}, Stanislaw Woźniak\textsuperscript{1}, Mathias Gehrig\textsuperscript{3}, Jules Lecomte\textsuperscript{4}, Axel von Arnim\textsuperscript{4}, Luca Benini\textsuperscript{2,5}, Davide Scaramuzza\textsuperscript{3}, Angeliki Pantazi\textsuperscript{1}

\textsuperscript{1}IBM Research – Zurich \textsuperscript{2}ETH Zurich \textsuperscript{3}University of Zurich \textsuperscript{4}fortiss GmbH \textsuperscript{5}Università di Bologna

Abstract

Optical flow provides information on relative motion that is an important component in many computer vision pipelines. Neural networks provide high accuracy optical flow, yet their complexity is often prohibitive for application at the edge or in robots, where efficiency and latency play crucial role. To address this challenge, we build on the latest developments in event-based vision and spiking neural networks. We propose a new network architecture, inspired by Timelens, that improves the state-of-the-art self-supervised optical flow accuracy when operated both in spiking and non-spiking mode. To implement a real-time pipeline with a physical event camera, we propose a methodology for principled model simplification based on activity and latency analysis. We demonstrate high speed optical flow prediction with almost two orders of magnitude reduced complexity while maintaining the accuracy, opening the path for real-time deployments.

1. Introduction

Optical flow is defined as an apparent motion of objects, edges and surfaces in a visual scene registered by the observer. It is caused by the relative motion between the observer and the scene and does not distinguish between actual motion in the visual scene and change in the observer’s pose. The applications of optical flow in the field of computer science include motion estimation and video compression [1, 7]. In machine perception as well as in robotics, optical flow is used for both object detection and tracking [3, 14, 27], robot navigation and even for control of micro air vehicles [9, 17, 31].

Deployment of optical flow in real-time robotic scenarios requires low-latency processing and energy efficiency. Existing algorithms usually calculate optical flow at discrete rates based on frames obtained from conventional cameras [13]. Neuromorphic dynamic vision sensors (DVS) operate similarly to the eye’s retina by providing a continuous stream of events representing brightness changes rather than absolute measurements at fixed time intervals [15]. Since optical flow computation relies on regions and time instants where brightness changes, DVS represents a viable alternative for fast optical flow prediction, demonstrated in recent works [6, 26]. Moreover, the sparsity of the events can be exploited by spiking neural networks (SNN) as opposed to artificial neural networks (ANNs). The advantage of SNNs deployed on neuromorphic hardware is low latency and energy efficiency coming from sparse computations [4].

Recently, researchers have presented an approach to produce sparse optical flow based on event data with SNNs [18]. However, there is a disconnect between large-scale architecture modelling and real-time deployments in efficient hardware. Here, we present a novel approach of a Timelens-like network for sparse optical flow predictions. Apart from surpassing the optical flow baseline in terms of the average endpoint error (AEE) [18], we also address the deployment aspect through systematic model reduction and demonstrate real-time operation with a physical DVS camera, as schematically illustrated in Fig. 1.

This paper makes the following contributions:

1. We design an optical flow architecture inspired by the Timelens architecture, enriched with spiking neurons operating with DVS-based inputs.
2. We surpass the state-of-the-art self-supervised optical
flow performance for SNNs and ANNs on the MVSEC dataset [33] for event-based vision, reducing the prediction error by 6.1% in spiking, 15.6% in analog-valued spiking, and 5.5% in non-spiking mode.

3. We propose a principled methodology, involving activity and latency analysis, for reducing the network size to fit into realistic real-time hardware constraints.

4. We demonstrate model reduction from 20.4M to 0.32M parameters with 0% penalty in error with regard to the prior art [18], enabling real-time operation with DVS inputs.

2. Related Work

2.1. Deep learning of SNNs

In recent years, SNN popularity in machine learning has been increasing owing to research advancements that enabled easy modelling and training in deep learning frameworks [23, 30]. Beyond the standard Leaky Integrate-and-Fire (LIF) model, an even wider variety of neuron-inspired spiking models has been explored. In particular, a framework around so-called Spiking Neural Unit (SNU) includes the plain SNU with LIF dynamics and typical axo-dendritic synapses, as well as its variants that model further biological aspects, such as axo-axonic and axo-somatic synapses in SNUo and SNUa, respectively. These variants demonstrated improvements in large-scale speech recognition models [5]. In the context of optical flow, modifications of a LIF implementation were also proposed and called ALIF, XLIF, and PLIF [18].

2.2. Architectures for optical flow

Successful training of neural networks relies on a proper loss definition, where historically supervised losses were used [13, 16, 28]. Due to challenges with obtaining a large number of high-quality labels, it is beneficial to reformulate the training in terms of a self-supervised loss [21, 32]. In [18], the optical flow prediction task was posed as a self-supervised contrast maximization problem. This training approach can be applied to popular network architectures for optical flow prediction that include EV-FlowNet [32] and FireNet [25]. State-of-the-art SNN implementations are based on their adaptation to inputs from event-based cameras and the operation with spiking neurons [18].

2.3. Timelens

The Timelens architecture [29] was proposed in the context of event-based video frame interpolation. The design itself has been inspired from the hourglass network with skip connections for frame-based video interpolation – a problem posed initially in [19]. A peculiarity of the network architecture is the reduction of the spatial dimensionality in the encoding part using a pooling operator rather than exploiting strided convolutions. Another feature is the bigger kernel sizes for the initial two convolutions compared with the rest of the encoding/decoding blocks.

3. Network model

We propose an architecture for prediction of optical flow based on SNNs receiving an event stream from DVS. Design choices, such as spatial down- and up-sampling, channel dimensions, kernel sizes and skip connections, are inspired by the Timelens network [29]. Our network is reformulated as an SNN by incorporating spiking spatial convolutions featuring stateful neural cells and layer recurrence. An overall architectural diagram is presented in Fig. 2 and the details are described in the following subsections.

3.1. Neuron models

First, we implement an SNN using state equations that describe the common neuroscientific LIF model in a form trainable within the realm of deep learning [18, 23, 30]:

\[ s_t = (1 - d)(Wx_t + Hy_{t-1}) + ds_{t-1}(1 - y_{t-1}) \]  \hspace{1cm} (1)

\[ y_t = h(s_t - v_{th}) \]  \hspace{1cm} (2)

where \( s_t \) is the state – membrane potential voltage of the neuron, \( W \) and \( H \) are the input and optional recurrent weights, respectively, \( d \) is membrane potential decay factor, \( y_t \) is the output, \( v_{th} \) is a firing voltage threshold, and \( h \) is the step activation function. The model is trainable with backpropagation-through-time assuming a smooth derivative of \( \text{arctan}(x, a) = 1/(1 + a \cdot x^2) \) for \( h \), with \( a = 10 \). Trainable parameters include \( W, H, d \) and \( v_{th} \). This neuronal model is our main focus and we will quantify architectures using it with the SNN prefix.

Secondly, we consider a more advanced biologically-inspired extension of the basic LIF – the so-called SNUo unit, which models the concept of axo-axonic synapses that enrich neuronal connectivity by modulating the neuronal outputs [5]. From implementation perspective, this leads to emission of sparse analog-valued spikes, or graded spikes in the nomenclature of Intel’s Loihi 2 implementation [24]. The equations of SNUo are [5]:

\[ s_t = g(Wx_t + Hy_{t-1} + ds_{t-1}(1 - y_{t-1})) \]  \hspace{1cm} (3)

\[ y_t = h(s_t - v_{th}) \]  \hspace{1cm} (4)

\[ y_t = o(W_a x_t + H_a y_{t-1} + b_a), \]  \hspace{1cm} (5)

where \( \tilde{y}_t \) is the unmodulated neuron output used for resetting the membrane potential, \( y_t \) is the modulated output propagating to downstream units, and \( g \) is an additional activation function that we set to leaky ReLU with a leak of 0.1. \( W_a, H_a \) and \( b_a \) are additional trainable parameters. We use
the \textit{sigmoid} function as activation $o$ for output modulation in Eq. 5 to mimic the inhibitory character of the axo-axonic synapses as suggested in [5]. We will quantify networks using this approach with the SNUo prefix.

Lastly, the benefits of neuromorphic internal dynamics were demonstrated also in the non-spiking mode: by operating with real-values in the so-called soft SNU (sSNU) approach [30]. The idea is to replace the step activation function $h$ with \textit{sigmoid} function in Eq. 2. As sSNU operates by continuously outputting real values, we will benchmark it against non-spiking baselines. We will quantify networks using this unit with the sSNU prefix.

3.2. Network structure

Spiking convolutions work similarly to conventional convolutions found in ANNs except for the neural dynamics applied to their outputs. The computed per pixel, per channel outputs of the 2D convolutions serve as input currents ($W_x t$ in Eq. 1) for the spiking neural units. Simultaneously, layer-wise recurrency ($H_{y_{t-1}}$ in Eq. 1) is an additional feature to capture temporal dependencies that is not always considered in SNN modelling. We explicitly mention whenever we do include this term.

The network structure is illustrated in Fig. 2. Its first stage comprises two spiking 2D convolutions expanding the $N_{in}$ input channels to 32 output channels featuring $7 \times 7$ kernels. While the spatial dimension is retained for the spiking convolution by using stride 1 and appropriate zero padding, spatial down-sampling is performed afterwards using 2D average pooling with kernel size $2 \times 2$. The remaining encoding parts of the network are five similar encoding blocks consisting of two spiking convolutions followed by pooling operators. The kernel sizes are $3 \times 3$, except for the first encoding block ($5 \times 5$). For each encoding block the number of output channels is doubled while the spatial resolution is halved. For the two spiking convolutions in each encoding block, we consider all combinations of layer-wise recurrency, as marked in Fig. 2.

For decoding, five identical decoding blocks are used. Each consists of 2D bilinear up-sampling by a factor of 2, followed by two spiking convolutions. The number of output channels gets halved with each decoding block and the convolutional kernel sizes are $3 \times 3$. Skip connections between each encoder/decoder pair of the same resolution provide values which are concatenated channel-wise before the second spiking convolution in each decoding block.

To obtain continuous optical flow values, the final layer is a $1 \times 1$ convolution with \textit{tanh} activation. This layer reduces 32 base channels to $N_{out} = 2$ channels representing the optical flow components $u$ and $v$ that correspond to horizontal and vertical optical flow magnitudes, respectively.

3.3. Input coding

The DVS event stream contains events of the form:

$$e_t = (x_i, y_i, t_i, p_i)$$  \hspace{1cm} (6)

where $x_i$ and $y_i$ represent the pixel coordinates, $t_i$ the timestamp and $p_i$ the ON/OFF polarity of the event. Different encoding strategies have been proposed to process the raw event stream data prior to feeding it into a neural network. Commonly used input coding techniques are the count encoding [22] and the voxel grid encoding [34], depicted in Fig. 3. The count encoding loses the temporal information of single events within the aggregation window. Events get

![Figure 3](image-url)
accumulated per pixel and per polarity for the entire window width. On the other hand, a voxel-based representation discretizes the time span of the aggregation window and uses temporal bi-linear interpolation to populate the bins with events. Polarity is not treated as a separate channel, but negative OFF events (-1) and positive ON events (+1) are summed in a single channel.

For our spiking architectures, we opted for the voxel grid input coding. The number of discrete time bins is an additional hyperparameter. Choosing the number of bins too high yields overly sparse inputs while for a low number of bins the encoding collapses to a count representation with a single channel. In the latter case, positive and negative events can annihilate each other leading to information loss. For our spiking network, performance peaked at six time bins \( N_{in} = 6 \). However, when operating in the non-spiking mode of sSNU, the count encoding with separate ON/OFF channels \( N_{in} = 2 \) performed better, so we use it for sSNU-based networks. To ensure a fair comparison, the aggregation window width is fixed and the set of encoded events is therefore the same for both encoding approaches.

3.4. Training setup

All models are trained in a self-supervised fashion on the UZH-FPV Drone Racing Dataset [11], using the approach and configurations from [18]. Specifically, contrast maximization loss is applied there to compensate the motion and predict optical flow from the input events. The loss is:

\[
L = L_{\text{contrast}}(t_{\text{ref}}) + L_{\text{contrast}}(t_{\text{ref}}^{bw}) + \lambda L_{\text{smoothing}}
\]  

(7)

where contrast maximization is performed in a forward \( (t_{\text{ref}}) \) as well as a backward \( (t_{\text{ref}}^{bw}) \) fashion w.r.t. the current reference time instance \( t_{\text{ref}} \).  \( L_{\text{smoothing}} \) is a Charbonnier smoothness prior [8] proposed in [32, 34] and \( \lambda = 0.001 \) is a balancing constant. Truncated back-propagation through time (TBPTT) is performed after every 10 forward passes.

In the original approach [18], the loss included different spatial resolutions of the optical flow maps. We analogously extended our architecture with 2D convolutions with \( \tanh \) activation to produce optical flow predictions of different resolutions at each decoding block. These intermediate optical flow maps are then up-sampled to the initial spatial dimension using nearest neighbour interpolation for the loss computation. Simultaneously, they are concatenated to the input channels of the subsequent decoding blocks.

However, in contrast to the prior work, we also considered an architecture with the loss applied only to the last output layer’s prediction. This approach is simpler, faster and turned out to be beneficial for our architecture.

4. Simulation results

The quantitative performance and generalization abilities of the trained models (self-supervised on the UZH-FPV Drone Racing Dataset) are evaluated on the MVSEC dataset [33] following the comparison approach from [18]. The predicted sparse optical flow is compared against the ground truth optical flow provided by [32]. The ground truth labels are available at timestamps corresponding with conventional camera’s frames and quantify the optical flow over one \( (dt = 1) \) or four \( (dt = 4) \) frames.

The well-established average end point error (AEE) in pixels is used to evaluate the four sequences of the dataset: outdoor_day1 (od1), indoor_flying1 (if1), indoor_flying2 (if2), indoor_flying3 (if3). For easier comparability, we introduce a weighted average endpoint error (WAEE) to combine the four metrics into a single scalar value:

\[
\text{WAEE} = (\frac{\text{W AEE}_{\text{od1}}}{w_{\text{od1}}} + \frac{\text{W AEE}_{\text{if1}}}{w_{\text{if1}}} + \frac{\text{W AEE}_{\text{if2}}}{w_{\text{if2}}} + \frac{\text{W AEE}_{\text{if3}}}{w_{\text{if3}}})/4,
\]  

(8)

where the four weights are based on the average AEE of the best-performing spiking architectures of the prior art [18] – see Supplementary Note 1 for the values for each \( dt \) setting.

Using the WAEE metric, we explored different configu-
Table 3. Evaluation on the MVSEC dataset for comparable models trained on UZH-FPV Drone Racing Dataset: AEE (the lower, the better ↓), the percentage of outliers %Out. (↓) per sequence, and the overall WAEE (↓) as defined in Eq. 8 as well as the average percentage of outliers %Out. (↓). Best scores are in bold, while runner-ups are underlined. Horizontal lines delimit the spiking and the non-spiking models.

<table>
<thead>
<tr>
<th></th>
<th>outdoor_day1</th>
<th>indoor_flying1</th>
<th>indoor_flying2</th>
<th>indoor_flying3</th>
<th>overall</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>AEE</td>
<td>%Out.</td>
<td>AEE</td>
<td>%Out.</td>
<td>AEE</td>
</tr>
<tr>
<td>LIF-EV-FlowNet [18]</td>
<td>0.53</td>
<td>0.33</td>
<td>0.71</td>
<td>1.41</td>
<td>1.44</td>
</tr>
<tr>
<td>XLIF-EV-FlowNet [18]</td>
<td>0.45</td>
<td><strong>0.16</strong></td>
<td>0.73</td>
<td>0.92</td>
<td>1.45</td>
</tr>
<tr>
<td>LIF-FireNet [18]</td>
<td>0.57</td>
<td>0.40</td>
<td>0.98</td>
<td>2.48</td>
<td>1.77</td>
</tr>
<tr>
<td>PLIF-FireNet [18]</td>
<td>0.56</td>
<td>0.38</td>
<td>0.90</td>
<td>1.93</td>
<td>1.67</td>
</tr>
<tr>
<td>our SNN-Timelens</td>
<td>0.44</td>
<td>0.18</td>
<td>0.70</td>
<td><strong>0.79</strong></td>
<td>1.30</td>
</tr>
<tr>
<td>our SNUo-Timelens</td>
<td>0.39</td>
<td>0.17</td>
<td><strong>0.60</strong></td>
<td>0.96</td>
<td><strong>1.17</strong></td>
</tr>
<tr>
<td>EV-FlowNet [18]</td>
<td>0.47</td>
<td>0.25</td>
<td>0.60</td>
<td><strong>0.51</strong></td>
<td>1.17</td>
</tr>
<tr>
<td>RNN-EV-FlowNet [18]</td>
<td>0.56</td>
<td>1.09</td>
<td>0.62</td>
<td>0.97</td>
<td>1.20</td>
</tr>
<tr>
<td>our sSNU-Timelens</td>
<td>0.36</td>
<td><strong>0.10</strong></td>
<td><strong>0.58</strong></td>
<td>0.56</td>
<td>1.19</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>indoor_flying4</th>
<th>overall</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>AEE</td>
<td>%Out.</td>
</tr>
<tr>
<td>LIF-EV-FlowNet [18]</td>
<td>0.50</td>
<td>1.09</td>
</tr>
<tr>
<td>XLIF-EV-FlowNet [18]</td>
<td>0.56</td>
<td>1.95</td>
</tr>
<tr>
<td>LIF-FireNet [18]</td>
<td>0.60</td>
<td>2.51</td>
</tr>
<tr>
<td>PLIF-FireNet [18]</td>
<td>0.60</td>
<td>2.51</td>
</tr>
<tr>
<td>our SNN-Timelens</td>
<td>0.44</td>
<td><strong>0.17</strong></td>
</tr>
<tr>
<td>our SNUo-Timelens</td>
<td>0.39</td>
<td>0.17</td>
</tr>
<tr>
<td>EV-FlowNet [18]</td>
<td>0.47</td>
<td>0.25</td>
</tr>
<tr>
<td>RNN-EV-FlowNet [18]</td>
<td>0.56</td>
<td>1.09</td>
</tr>
<tr>
<td>our sSNU-Timelens</td>
<td>0.36</td>
<td><strong>0.10</strong></td>
</tr>
<tr>
<td></td>
<td>indoor_flying4</td>
<td>overall</td>
</tr>
<tr>
<td></td>
<td>AEE</td>
<td>%Out.</td>
</tr>
<tr>
<td>LIF-EV-FlowNet [18]</td>
<td>0.50</td>
<td>1.09</td>
</tr>
<tr>
<td>XLIF-EV-FlowNet [18]</td>
<td>0.56</td>
<td>1.95</td>
</tr>
<tr>
<td>LIF-FireNet [18]</td>
<td>0.60</td>
<td>2.51</td>
</tr>
<tr>
<td>PLIF-FireNet [18]</td>
<td>0.60</td>
<td>2.51</td>
</tr>
<tr>
<td>our SNN-Timelens</td>
<td>0.44</td>
<td><strong>0.17</strong></td>
</tr>
<tr>
<td>our SNUo-Timelens</td>
<td>0.39</td>
<td>0.17</td>
</tr>
<tr>
<td>EV-FlowNet [18]</td>
<td>0.47</td>
<td>0.25</td>
</tr>
<tr>
<td>RNN-EV-FlowNet [18]</td>
<td>0.56</td>
<td>1.09</td>
</tr>
<tr>
<td>our sSNU-Timelens</td>
<td>0.36</td>
<td><strong>0.10</strong></td>
</tr>
</tbody>
</table>

Table 3. Evaluation on the MVSEC dataset for comparable models trained on UZH-FPV Drone Racing Dataset: AEE (the lower, the better ↓), the percentage of outliers %Out. (↓) per sequence, and the overall WAEE (↓) as defined in Eq. 8 as well as the average percentage of outliers %Out. (↓). Best scores are in bold, while runner-ups are underlined. Horizontal lines delimit the spiking and the non-spiking models.

The results of the layer recurrency in the convolutional blocks, visualized in Fig. 2. As each block comprises two spiking convolutions, there are four different combinations of recurrent (R) and feed-forward (F) convolutions: R/F, F/R, R/R and F/F. Table 1 reports the results in terms of WAEE and the average percentage of outliers %Out. When operating in the spiking mode, having one convolution with layer recurrency per block is favourable. In particular, best performance is achieved with recurrent layers in the first convolution (R/F). On the contrary, in the context of non-spiking mode of sSNU, double layer recurrency (R/R) is beneficial. We use these best configurations for the final models.

We also evaluated an implementation of multi-resolution loss, described in the training section. For both settings of dt = 1 and dt = 4, the reported WAEE values in Table 2 demonstrate that using the simpler setup of the loss applied only at the last layer is preferred for our architecture. A possible interpretation of the observed deterioration is that the multi-layer loss function trains the deeper decoders to encode down-sampled optical flow rather than to develop higher-level features. Furthermore, such a formulation is inconsistent with the ultimate task of the network, which is predicting high-resolution optical flow at the last layer rather than outputting the flow predictions at multiple intermediate stages. Imposing a loss only on the last layer, omits this restriction. We use this approach for all our models.

The resulting AEEs, WAEEs and outlier percentages (AEE > 3 pixels) for our Timelens-based architecture with spiking (SNN), analog-valued spiking (SNUo) and non-spiking (sSNU) units are reported in Table 3. Our model is compared with the state-of-the-art spiking and non-spiking architectures trained in the identical self-supervised setting [18]. For an extended comparison with EV-FlowNet [32,34] and Hybrid-EV-FlowNet [20] that use different training datasets and setups, see Supplementary Note 2.

For spiking neural networks, our SNN-Timelens surpasses the performance of the LIF- and XLIF-EV-FlowNet by 9.7%, 6.7% with regard to WAEE and lowers the percentage of outliers %Out. by 30.5%, 24.1% for dt = 1, respectively. As the improvement over the XLIF-EV-FlowNet is 5.6% for dt = 4, the average prediction error is reduced by 6.1%. Table 3 shows that our SNNs are not only better on average, but outperform the comparable state-of-the-art on each MVSEC sequence for dt = 1 and dt = 4.

Operating with analog-valued spikes, SNUo-Timelens achieves a further substantial reduction in WAEE: 18.3%
and 15.6% vs. LIF- and XLIF-EV-FlowNet for \(dt = 1\), respectively. Since the improvement over XLIF-EV-FlowNet for \(dt = 4\) is the same, an average decrease is 15.6%. It is also remarkable that SNUo-Timelens demonstrates on-par performance with the best non-spiking self-supervised prior-art for comparable training configurations.

Lastly, the best performing model is the sSNU-Timelens incorporating neuromorphic dynamics into the non-spiking mode of operation. Despite featuring 8.5% less parameters than the best baseline EV-FlowNet (32.9M), our sSNU-TimeLens (30.1M) outperforms it with regard to WAEE for both \(dt = 1\), \(dt = 4\) by 6.4%, 4.1%, respectively. The average improvement over the state-of-the-art for comparable non-spiking models therefore equals 5.5%.

5. Model reduction for real-time operation

The state-of-the-art models listed in Table 3 involve tens of millions of parameters and are executed on high-end GPUs. To close the gap between large-scale architecture modelling and real-time deployment, model reduction is required. We propose a principled approach for model reduction that includes analysis of the network activity and of the relationships between the number of parameters and inference speed at different stages of the architecture.

We focus our exploration on the SNN-Timelens that could benefit the most from efficient implementation on SNN chips, such as TrueNorth [2], Loihi [10] or Kraken’s SNE [12], that support the LIF equations used in the SNU. If support for analog-valued spikes increases, as in Loihi 2, the SNUo-Timelens architecture could become appealing.

5.1. Spiking activity analysis

A spiking activity analysis has been conducted to obtain potential information about the importance of different network building blocks. For a test sequence of the MVSEC dataset, the fraction of neurons that produced spikes was registered for each network layer: input layer, initial convolutional layers, encoding layers \(s[i]\), decoding layers \(u[i]\) and the final prediction layer. Fig. 4 shows the spiking activity in SNN-Timelens architecture with 5 encoding/decoding blocks (left) compared with a network reduced to 3 encoding/decoding blocks (right). In the following we will refer to the number of encoding/decoding blocks as the number of stages of the Timelens model.

In general, the fraction of non-zero outputs, which corresponds to the fraction of neurons that spike, is almost constant until time step 210. At this time step, the drone in the DVS recording lifts off and the incoming events actually come from movement rather than static noise. The spiking activity for all layers fluctuates between 0 and 0.5 when optical flow is predicted due to the actual movement. It has to be noted that the activity of the last layer is 1.0 for all times since the final prediction layer does not feature a \(\text{step}\) but rather a continuous \(\tanh\) activation function.

For the bigger model comprising 5 stages the fraction of non-zero outputs does not vary at all for the deep encoding \(s3 - s5\) and decoding \(u1 - u3\) layers. However, evaluation of the gradients indicates that the weights get updated during training. The question therefore arises whether these deep layers are crucial for the overall model performance. Reducing the number of stages from 5 to 3 shows indeed almost on par performance, only 2.6 % WAEE drop on MVSEC, while the spiking activity varies for all layers. The smaller model features only 1.75M parameters, which is 14.5 times less than the initial SNN architecture with 25.35M. The constant spiking activity for the layers can be interpreted as a quasi-identity mapping between early encoding and late decoding layers. Thus, dropping these layers tends to have a minor effect on the network capabilities.
5.2. Network profiling

In deep CNNs there is no simple linear relationship between the number of parameters and the inference latency. Therefore, we profiled the contributions of the components of the model to assess how the number of stages (encoding/decoding blocks) and the size of convolution impacts the inference frequency in frames per second (fps). Model performance is monitored throughout the process to find a balance between speed and quality of the predicted optical flow. The fps values are calculated from timings of 100 forward passes on 128×128 DVS inputs using Pytorch code executed on a single core of Intel Core i7 2.6GHz CPU.

Reducing channels. Network profiling has revealed that the first convolution and the first encoding block are particularly costly in terms of computations. On one hand this is due to large spatial input dimension, on the other hand it is influenced by the big convolutional kernels (7×7 and 5×5). Nevertheless, decreasing the number of output channels effectively reduces the computational costs. Fig. 5 illustrates a trade-off between the number of channels and performance in terms of WAAEE and fps for the SNN-Timelens model with 5 stages. Note the non-linear relationship between convolutional channels and network parameters.

Reducing stages. The spiking analysis showed that less than 5 stages, e.g. 3 stages, are sufficient to obtain reasonable optical flow predictions. Table 4 extends the analysis, reporting the WAAEE (dt = 1), the number of network parameters and model inference frequency for different number of channels and stages. Comparing the WAAEE between 5 and 2 stages, we observe minor performance degradation: 0.84 versus 0.86. The 2-stage model comes with 44.4 times less parameters and increases the evaluation frequency by 93.2%. For further speedup, the number of channels of the 2-stage SNN-Timelens model can be decreased at the cost of degrading performance in terms of WAAEE.

6. Model reduction results

The comparison of our architecture before and after reduction is presented for a set of selected configurations in Fig. 6. While our initial SNN-Timelens featured 5 stages with 32 channels and used 25.35M parameters, our model after reduction features only 2 stages with 32 channels and 0.57M parameters, thus reducing the number of trainable parameters by a factor of 44.4. It involves a trade-off in

<table>
<thead>
<tr>
<th># channels</th>
<th>32</th>
<th>28</th>
<th>24</th>
<th>20</th>
<th>16</th>
<th>12</th>
</tr>
</thead>
<tbody>
<tr>
<td>5 stages</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>WAAEE</td>
<td>0.84</td>
<td>0.89</td>
<td>0.89</td>
<td>0.93</td>
<td>0.94</td>
<td>1.02</td>
</tr>
<tr>
<td>parameters</td>
<td>25.3</td>
<td>19.4</td>
<td>14.3</td>
<td>9.9</td>
<td>6.3</td>
<td>3.6</td>
</tr>
<tr>
<td>frequency</td>
<td>5.9</td>
<td>7.3</td>
<td>9.9</td>
<td>12.8</td>
<td>19.4</td>
<td>26.1</td>
</tr>
<tr>
<td>3 stages</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>WAAEE</td>
<td>0.86</td>
<td>0.88</td>
<td>0.90</td>
<td>0.91</td>
<td>0.93</td>
<td>1.00</td>
</tr>
<tr>
<td>parameters</td>
<td>1.75</td>
<td>1.34</td>
<td>0.98</td>
<td>0.68</td>
<td>0.44</td>
<td>0.25</td>
</tr>
<tr>
<td>frequency</td>
<td>9.8</td>
<td>11.1</td>
<td>14.8</td>
<td>17.2</td>
<td>26.2</td>
<td>32.7</td>
</tr>
<tr>
<td>2 stages</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>WAAEE</td>
<td>0.86</td>
<td>0.89</td>
<td>0.90</td>
<td>0.93</td>
<td>0.94</td>
<td>1.10</td>
</tr>
<tr>
<td>parameters</td>
<td>0.57</td>
<td>0.44</td>
<td>0.32</td>
<td>0.23</td>
<td>0.15</td>
<td>0.08</td>
</tr>
<tr>
<td>frequency</td>
<td>11.4</td>
<td>13.2</td>
<td>18.7</td>
<td>21.4</td>
<td>30.1</td>
<td>36.3</td>
</tr>
</tbody>
</table>

Table 4. Impact of convolutional channels count: WAAEE, number of network parameters (in millions [M]), and inference frequency (in frames per second [fps]) of our Timelens-based SNNs for 5, 3 and 2 stages (encoding/decoding blocks).
terms of WAEE performance that degrades by just 2.6% (0.84 versus 0.86). Remarkably, it still remains better than the prior state-of-the-art large models of LIF- and XLIF-EV-FlowNet (20.4M) with WAEE 0.90 and 0.93, respectively. Note that to match prior art performance (WAEE 0.90), our SNN-Timelens needs only 2 stages with 24 channels (0.32M), featuring 63.75 times less parameters.

6.1. Qualitative results

For qualitative performance assessment and validation of the generalization ability of the last proposed network with 2 stages and 24 channels, a complete real-time pipeline was implemented to process the event stream of a DVS128 camera from iniVation AG. Fig. 7 shows the optical flow predictions of different hand movements in front of the DVS. While color-coding is used to encode the optical flow, additionally a sparse arrow grid is superimposed to the optical flow for instant intuitive validation of the predictions. Arrow angles and magnitudes represent direction and magnitude of the biggest optical flow within a local $10 \times 10$ neighborhood of pixels, respectively.

The predicted flow in Fig. 7 looks reasonable and coincides with the expected dislocations caused by the moving hand. Linear motion is correctly captured (left plots) and the model generalizes well to more challenging scenarios such as rotating or approaching hand (right plots).

7. Conclusion

In this work we proposed a neuromorphic solution for optical flow estimation comprising an event camera combined with a Timelens-inspired architecture. We demonstrated SNN, SNUo and sSNU versions of our architecture, operating with different biologically inspired neuron models. By tuning the architectural design, the event encoding, the placement of recurrent connections, and the loss function formulation, we improved the performance in comparison with prior art models on the MVSEC dataset. Our architecture surpassed both SNN and ANN baselines when operating in spiking and real-valued modes, respectively. Remarkably, when operating with analog-valued spikes, it demonstrated performance comparable to the ANN baseline. Furthermore, a principled model reduction approach was proposed to meet realistic real-time hardware constraints. Our SNN-Timelens model reduced to 0.32M parameters achieves WAEE on-par with the state-of-the-art while decreasing the number of parameters by almost two orders of magnitude. Finally, a real-time pipeline was demonstrated with a physical DVS camera. Future work includes deployment of the proposed architecture on a neuromorphic SNN chip to further decrease the latency and increase energy efficiency.

8. Acknowledgements

The research was carried out in collaboration with the IBM and fortiss Center for AI (C4AI). We thank its team for discussions and assistance. The research at fortiss was supported by the HBP Neurorobotics Platform funded through the European Union’s Horizon 2020 Framework Program for Research and Innovation under the Specific Grant Agreements No. 945539 (Human Brain Project SGA3).
References


[22] Ana I. Maqueda, Antonio Loquercio, Guillermo Gallego, Narciso Garcia, and Davide Scaramuzza. Event-based vision...


Supplementary Notes for
Neuromorphic Optical Flow and Real-time Implementation with Event Cameras

Yannick Schnider¹,², Stanisław Woźniak¹, Mathias Gehrig³, Jules Lecomte⁴, Axel von Arnim⁴, Luca Benini²,⁵, Davide Scaramuzza³, Angeliki Pantazi¹
¹IBM Research – Zurich ²ETH Zurich ³University of Zurich ⁴fortiss GmbH ⁵Università di Bologna

1. WAEE metric definiton

The weighted average end point error (WAEE) combines the AEEs of the four sequences:

- outdoor_day1 (od1)
- indoor_flying1 (if1)
- indoor_flying2 (if2)
- indoor_flying3 (if3)

and is defined as:

\[
\text{WAEE} = \left( \frac{\text{AEE}_{\text{od1}}}{w_{\text{od1}}} + \frac{\text{AEE}_{\text{if1}}}{w_{\text{if1}}} + \frac{\text{AEE}_{\text{if2}}}{w_{\text{if2}}} + \frac{\text{AEE}_{\text{if3}}}{w_{\text{if3}}} \right) / 4.
\]

The weights \( w_{\text{od1}} \), \( w_{\text{if1}} \), \( w_{\text{if2}} \), \( w_{\text{if3}} \) are the sequence specific average AEEs of the spiking EV-FlowNet variants: LIF, ALIF, PLIF and XLIF for the modes \( dt = 1 \) and \( dt = 4 \) published in [1]:

\[
dt = 1:
\begin{align*}
    w_{\text{od1}} &= (0.53 + 0.57 + 0.60 + 0.45)/4 = 0.5375 \\
    w_{\text{if1}} &= (0.71 + 1.00 + 0.75 + 0.73)/4 = 0.7975 \\
    w_{\text{if2}} &= (1.44 + 1.78 + 1.52 + 1.45)/4 = 1.5475 \\
    w_{\text{if3}} &= (1.16 + 1.55 + 1.23 + 1.17)/4 = 1.2775
\end{align*}
\]

\[
dt = 4:
\begin{align*}
    w_{\text{od1}} &= (2.02 + 2.13 + 2.24 + 1.67)/4 = 2.0150 \\
    w_{\text{if1}} &= (2.63 + 3.81 + 2.80 + 2.72)/4 = 2.9900 \\
    w_{\text{if2}} &= (4.93 + 6.40 + 5.21 + 4.93)/4 = 5.3675 \\
    w_{\text{if3}} &= (3.88 + 5.53 + 4.12 + 3.91)/4 = 4.3600
\end{align*}
\]

(1)

2. Additional comparison

Table 1 includes an extended comparison with additional prior art non-spiking models. In particular, EV-FlowNetPM⁴ was trained in comparable setting to ours, but used a photometric loss (PM). The results had been only reported for \( dt = 1 \) mode. Furthermore, several prior art architectures were trained in a different setup using directly the MVSEC dataset, as opposed to our architectures that were trained on the UZH-FPV Drone Racing Dataset and evaluated on the MVSEC dataset. Results for models trained directly on MVSEC, delimited by dashed lines, include:

- EV-FlowNetPM,MVSEC⁴, trained in a self-supervised manner with the photometric loss (PM),
- EV-FlowNetCM,MVSEC⁵, trained in a self-supervised manner with a contrast maximisation loss (CM),
- Hybrid-EV-FlowNetMVSEC², trained in a self-supervised manner with the photometric loss.

Considering the extended comparison with non-spiking ANN prior art models, the EV-FlowNet⁴ CM,MVSEC is the best performance on all MVSEC sequences for \( dt = 1 \) with regard to WAEE and percentage of outliers. Its WAEE of 0.67 is 8.2% lower than 0.73 of our sSNU-Timelens. In turn, the Hybrid-EV-FlowNet⁵ MVSEC[2] is outperformed by our sSNU-Timelens by 26.0% (0.73 vs. 0.92).

However, when evaluating in mode \( dt = 4 \), the Hybrid-EV-FlowNet⁵ MVSEC[2] yields the best overall performance with a WAEE of 0.68 compared to our sSNU-Timelens with 0.71 (+4.4%). The sSNU-Timelens shows on par performance in terms of WAEE with the EV-FlowNet⁴ CM,MVSEC⁵ (also 0.71) in this mode.

In summary, the EV-FlowNet⁴ CM,MVSEC[5] and the Hybrid-EV-FlowNet⁵ MVSEC[2] perform best for MVSEC evaluations with \( dt = 1 \) and \( dt = 4 \), respectively. Remarkably, our sSNU-Timelens is a runner-up in both cases, despite being trained without access to the examples from the MVSEC dataset.
### Table 1. Extended evaluation on MVSEC: AEE (the lower, the better ↓), the percentage of outliers % Out. (↓) per sequence, and the overall W AEE (↓) as defined in Eq. 1 as well as the average percentage of outliers % Out. (↓). Best scores are in bold, while runner-ups are underlined. Horizontal lines delimit the spiking and the non-spiking models. Dashed line delimits not directly comparable prior art setups.

<table>
<thead>
<tr>
<th>Method</th>
<th>outdoor_day</th>
<th>indoor_flying</th>
<th>indoor_flying</th>
<th>indoor_flying</th>
<th>overall</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>AEE</td>
<td>% Out.</td>
<td>AEE</td>
<td>% Out.</td>
<td>AEE</td>
</tr>
<tr>
<td>LIF-EV-FlowNet [1]</td>
<td>0.53</td>
<td>0.33</td>
<td>0.71</td>
<td>1.41</td>
<td>1.44</td>
</tr>
<tr>
<td>XLIF-EV-FlowNet [1]</td>
<td>0.45</td>
<td><strong>0.16</strong></td>
<td>0.73</td>
<td>0.92</td>
<td>1.45</td>
</tr>
<tr>
<td>LIF-FireNet [1]</td>
<td>0.57</td>
<td>0.40</td>
<td>0.98</td>
<td><strong>2.48</strong></td>
<td>1.77</td>
</tr>
<tr>
<td>PLIF-FireNet [1]</td>
<td>0.56</td>
<td>0.38</td>
<td>0.90</td>
<td>1.93</td>
<td>1.67</td>
</tr>
<tr>
<td>our SNN-Timelens</td>
<td>0.44</td>
<td>0.18</td>
<td>0.70</td>
<td><strong>0.79</strong></td>
<td>1.30</td>
</tr>
<tr>
<td>our SNUo-Timelens</td>
<td><strong>0.39</strong></td>
<td>0.17</td>
<td><strong>0.64</strong></td>
<td>0.96</td>
<td><strong>1.17</strong></td>
</tr>
<tr>
<td>EV-FlowNet [3]</td>
<td>0.53</td>
<td>0.33</td>
<td>0.71</td>
<td>1.41</td>
<td>1.44</td>
</tr>
<tr>
<td>EV-FlowNet [1]</td>
<td>0.45</td>
<td>0.16</td>
<td>0.73</td>
<td>0.92</td>
<td>1.45</td>
</tr>
<tr>
<td>RNN-EV-FlowNet [1]</td>
<td>0.56</td>
<td>0.40</td>
<td>0.98</td>
<td><strong>2.48</strong></td>
<td>1.77</td>
</tr>
<tr>
<td>our SNN-Timelens</td>
<td>0.44</td>
<td>0.18</td>
<td>0.70</td>
<td><strong>0.79</strong></td>
<td>1.30</td>
</tr>
<tr>
<td>our SNUo-Timelens</td>
<td><strong>0.39</strong></td>
<td>0.17</td>
<td><strong>0.64</strong></td>
<td>0.96</td>
<td><strong>1.17</strong></td>
</tr>
</tbody>
</table>

### References


