Event-based Temporally Dense Optical Flow Estimation with Sequential Neural Networks

Wachirawit Ponghiran, Chamika Mihiranga Liyanagedera and Kaushik Roy
Purdue University, West Lafayette, IN 47907, USA

Abstract

Event-based optical flow estimation techniques have been recently receiving much attention as they process temporally rich inputs generated by event cameras. Unlike traditional frame-based cameras that sample light intensity at a fixed interval, event-based cameras record the changes in intensity (or events) allowing them to operate at a much higher temporal resolution and without motion blur. Past studies have investigated several gradient-based learning methods to train neural networks for predicting optical flow from an event stream. However, they do not utilize the fast data rate of event data streams and rely on a spatio-temporal representation constructed from a collection of events over a fixed period of time (often between two grayscale frames). As a result, optical flow is only evaluated at a frequency much lower than the rate data is produced by an event-based camera, leading to a temporally sparse optical flow estimation. To predict temporally dense optical flow, we cast the problem as a sequential learning task and propose a training methodology to train sequential networks for continuous prediction on an event stream. We propose two types of networks: one focused on performance and another focused on compute efficiency. We first train long-short term memory networks (LSTMs) on the DSEC dataset and demonstrated 10× temporally dense optical flow estimation over existing flow estimation approaches. The additional benefit of having a memory to draw long temporal correlations back in time results in a 19.7% improvement in flow prediction accuracy of LSTMs over similar networks with no memory elements. We subsequently show that the inherent recurrence of spiking neural networks (SNNs) enables them to learn and estimate temporally dense optical flow with 31.8% lesser parameters than LSTM, but with a slightly increased error. This demonstrates potential for energy-efficient implementation of fast optical flow prediction using SNNs.
1 Introduction

Optical flow estimation is a core problem in computer vision that evaluates the motion of each pixel between any two consecutive images captured by a frame-based camera. Optical flow information enables an observer to visualize a motion field which is useful for numerous applications such as object trajectory prediction [1], robotic control [2], and autonomous driving [3]. The problem has been traditionally addressed using various classical computer vision techniques like correlation-based [4], block-matching [5] and energy minimization-based [6] techniques, but their computational costs have shown to be prohibitively expensive for real-time applications. Neural Network (NNs) based techniques for optical flow prediction [7–9] have been proposed and remain a popular low-cost computing method. Generally, NN based implementations receive two consecutive images taken by a frame-based camera as inputs and predict the optical flow that best warps important features from one image to another. However, due to the limited dynamic range of such frame-based cameras, the performance of the aforementioned techniques may be affected by motion blur or temporal aliasing [10].

Methods to estimate optical flow from event-camera outputs offer a promising alternative to the frame-based approaches [11–17]. An event-based camera logs an intensity change at each pixel (so-called events) rather than an actual light intensity for a fixed duration. Thus, an event-based camera can generate a stream of events at high temporal resolution as illustrated in Fig. 1(a). This makes event-based optical flow estimation less susceptible to motion blur and more suitable for dynamic scenes. Nonetheless, being able to effectively extract information from a continuous event stream is a challenging task. An event-based camera outputs events at a fast rate but in an asynchronous and noisy manner. To ensure high fidelity of the inputs to NNs, existing works collect events over a fixed period (often a duration between two consecutive grayscale frames) and construct a spatio-temporal representation for optical flow estimation. Hence, optical flow is evaluated at a speed slower than the rate events are produced by an event camera as illustrated in Fig. 1(b). Evaluating optical flow at a faster rate can be crucial for certain applications, such as dodging an obstacle during navigation [18], where fast reaction time is essential.

To predict temporally dense optical flow, we cast the optical flow estimation problem as a sequential learning task. This allows us to consider the event stream as a single contiguous input over time rather than a sequence of independent inputs. In other words, we enable optical flow to be evaluated at a much higher frequency as illustrated in Fig. 1(c). The proposed NNs are trained not only to extract important features from a given input but also to remember useful information to be used for future optical flow estimations (see arrows between NN blocks in Fig. 1(c)). For this task, we employ two NN implementations, each with internal states for learning. The first implementation is a long-short term memory network (LSTM) trained to estimate optical flow in a supervised manner. To address the inefficiency of LSTMs in handling discrete and synchronous event streams, we present a spiking neural network (SNN) for
Figure 1: (a) Comparison between outputs of a traditional frame-based and event-based camera. (b) Existing neural networks (NNs) typically rely on a collection of events for optical flow prediction. (c) The proposed NNs process each event count individually and combine new information with past information for instantaneous optical flow estimation without delay. Arrows between NN blocks indicate information from the past time-step is carried over to the future time-step for NN computation.
optical flow estimation as our second approach. SNNs are biologically inspired and known to operate in an event-driven manner. Thus, SNNs can be mapped into event-driven hardware allowing an end-to-end event-driven framework for flow estimation.

Steps to train NNs for continuous optical flow predictions are, however, not straightforward. A proper encoding scheme must be adapted to deliver event information at each time step to the input of the NNs. For this purpose, we use per-pixel event count obtained through simple aggregation over each time period. Temporal information of the data is implicitly encoded in the order that the event counts are fed to NNs. We later show that this simple input representation is sufficient to obtain improved optical flow estimation with the proposed network architecture. Another issue with sequential learning problems is the assumption that the sequential input is of limited length. It is slightly different in our case since we consider an event stream to be a long contiguous input during inference. We focus on a continuous optical flow estimation from an event stream without a network reset. Note, a reset implies losing event information that the NN have processed in the past. We show that typical sequential learning approaches do not train NNs to perform well on continuous flow estimation and propose modifications to the training methodology to address this issue. The proposed modification allows NNs to learn to ignore information from older events and consider more recent events for optical flow estimation.

Overall, our contributions can be summarized as follows:

1. We cast optical flow estimation as a sequential learning problem to achieve temporally dense optical flow prediction. We train sequential neural networks to produce 10× more temporally dense flow estimation over existing approaches.

2. We show that typical sequential training approaches do not translate well for continuous flow estimation. Our proposed modifications to the training methodology lead to a 19.7% improvement in the prediction accuracy of the LSTM network over a similar model without internal states. The improvement comes from the ability to draw longer temporal correlations from an event stream.

3. We propose an SNN architecture that is more efficient than LSTM networks in handling asynchronous event streams. SNNs can be mapped into event-driven hardware geared towards handling event data. The proposed SNN predicts temporally dense optical flow with acceptable quality using 31.8% fewer parameters and at least 31.1% fewer multiplication operations than the LSTM network.
2 Background

2.1 Network Architectures for Sequential Learning

Sequential learning tasks are a class of problems where information is received through multiple episodes over time. In sequential learning, NNs are trained to extract and retain important information at each time step for future predictions (e.g., optical flow). This complicates the underlying training methodology, since NNs would require memory elements to retain knowledge from the past. In this work, we cast temporally dense optical flow estimation as a sequential problem through LSTMs and SNNs. We consider LSTMs for this task owing to their generalization capabilities and SNNs for their event-driven nature and high fidelity with event data. The operation of LSTM and SNN at any particular time \( t \) can be visualized in the form of a computational graph as shown in Fig. 2(a).
2.1.1 Long-Short Term Memory Networks (LSTMs)

LSTMs are a type of NNs with internalized memory that have demonstrated exceptional generalization capabilities across various sequential learning problems \[19–21\]. Their internal state typically referred to as the cell state \((c_t)\) is specifically designed to avoid the vanishing gradient problem. Cell state runs straight through an LSTM with minimal linear interactions as illustrated in Fig. 3(a), thus avoiding encoded information from the past being disrupted while sustaining gradient flow during back-propagation.

Whenever an LSTM cell receives an input \((x_t)\) at time \(t\), it computes an output known as the hidden state \((h_t)\) and the internal state \((c_t)\) through various gating mechanisms as follows. First, the forget gate (see left dashed box in Fig. 3(a)) controls how much of the previous cell state \(c_{t-1}\) is retained by deriving a scale factor \(f_t\) (valued between 0 and 1) based on the input and the previous hidden state \(h_{t-1}\). Candidate and input gates (see middle box in Fig. 3(a)) then calculates the contribution from the input to the internal state and combines it with the output of the forget gate to obtain the new cell state. Lastly, the output gate controls the amount of information carried from the new cell state to the LSTM output (hidden state). The dynamic of a LSTM cell can be expressed mathematically as follows:

\[
\begin{align*}
    f_t &= \sigma(W_f h_{t-1} + W_f x_t + b_f) \\
    i_t &= \sigma(W_i h_{t-1} + W_i x_t + b_i) \\
    \hat{c}_t &= \tanh(W_c h_{t-1} + W_c x_t + b_c) \\
    c_t &= f_t \odot c_{t-1} + i_t \odot \hat{c}_t \\
    o_t &= \sigma(W_o h_{t-1} + W_o x_t + b_o) \\
    h_t &= o_t \odot \tanh(c_t)
\end{align*}
\]

where \(W\) and \(b\) are the weight and bias of each different gate. \(\odot\) signifies an element-wise multiplication.

2.1.2 Spiking Neural Networks (SNNs)

Spiking neurons are artificial neurons that are inspired by biological neurons in nature. They display several unique characteristics that make them suitable for real-time applications. Artificial spiking neurons communicate sparsely through binary signals (so-called spikes) that resemble electric pulses transmitted by the biological neurons. This communication scheme simplifies hardware implementations of SNNs and enables their computations to be done efficiently in an event-driven manner \[22–24\]. In addition, their event-driven nature makes them ideal for handling asynchronous data generated by event-based sensors. Spiking neurons also have internal states which are useful for sequential learning.

A spiking neuron has two internal states referred to as the synaptic current \((c_t)\) and the membrane potential \((v_t)\) \[25\]. The synaptic current is increased by an input coming into the neuron modulated by a synaptic weight. Increasing synaptic current then leads to a rise in the membrane potential. The neuron
generates an output (or a spike) when the membrane potential exceeds a defined threshold. Unfortunately, the traditional spiking neuron model is known to suffer from the vanishing gradient problem. However, it is possible to overcome this problem by slightly modifying the spiking neuron model as explained in [26]. We use this improved spiking neuron model illustrated in Fig. 3(b). The dynamics of this spiking neuron are governed by the following equations:

\[
\begin{align*}
    f_t &= \sigma(W_fx_t + b_f) \\
    i_t &= \sigma(W_ix_t + b_i) \\
    c_t &= f_t \odot c_{t-1} + i_t \odot (W_c x_t + b_i) \\
    v_t &= v_{t-1} - y_{t-1} + \tanh(c_t) \\
    y_t &= \text{thres}(v_t)
\end{align*}
\]

where \(y_t\) represents the output of a spiking neuron at time \(t\). The improved spiking neuron model is inspired by LSTM, thus it bears similar concept to that of forget (\(f_t\)) and input (\(i_t\)) gates. The output (\(y_t\)) of the improved model is not constrained to a binary value, but can have multi-bit resolution.

### 2.2 General Method to Train Sequential Networks

To understand the training methodology, we refer to the computational graphs of LSTM and SNN in Fig. 2. Because internal states of LSTM and SNN (\(state_t\)) are computed based on a new input (\(input_t\)) and their state values from the previous time-step (\(state_{t-1}\)), we can utilize the same computational graph to derive new internal states recursively. Hence, back-propagation through time (BPTT) algorithm can be applied to compute gradients for training these models. For this, the operations of the LSTM and SNN are unfolded in time by creating several copies of the NN and treating them as a feed-forward network with tied weights. Fig. 2(b) shows the flow of LSTM and SNN after an unrolling. Error is computed in a supervised manner and the gradient is propagated backward through each time-step after completing a forward pass. We overcome the non-differentiability of the SNN threshold function by substituting the ill-defined differential with a straight-through estimator [27].

### 3 Method

#### 3.1 Event Representation for Optical Flow Estimation

Selecting a proper event representation for optical flow estimation is a challenging task as an event-based camera asynchronously reports changes in the light intensity (\(I_t\)) at every pixel on the sensor array. For each pixel, an event camera can generate a negative event or a positive event. The positive event is generated whenever the brightness increases beyond a predefined threshold (\(\theta^+\)) as described by the following equation:

\[
\log(I_t/I_{t-1}) \geq \theta^+
\]
Likewise, the negative event is generated when the brightness decreases beyond a different threshold $\theta^-$. Hence, each event corresponds to a time ($t$), pixel location ($x,y$) and polarity of change ($p$). Since the goal of optical flow estimation is to produce image-like output that indicates the flow magnitude in $x$-and $y$-directions, existing approaches utilize a convolutional layer to draw spatial correlations between nearby pixels. Hence, the common practice is to structure event information as frames with a fixed number of channels before convolution is applied.

Prior works proposed different methods to construct this spatio-temporal representation from a collection of events. One common approach encodes the average timing or the most recent timing of events at every pixel into one of the channels to capture some temporal information [11–13]. Another common approach divides events into multiple partitions with the same number of events. Then, per-pixel event count from each partition is calculated to form a multi-channel input [15–17]. The issue with these input encoding schemes is that an inference can only be made once the entire sequence of event data is available. For instance, suppose that we want to represent events received over a duration between $t=16$ and $t=20$ from a continuous event stream as depicted in Fig. 1(c). At $t=18$, events cannot be divided into equivolume partitions and translated to a spatio-temporal representation since the total number of events that arrives during a whole duration is not yet available.

To enable instantaneous computing from events in a smaller interval, we feed per-pixel event count as an input to NNs. This representation can be obtained through simple aggregation over each time period. Since our proposed NNs process input sequentially, temporal information of events is implicitly encoded in the order that the event counts are fed to the NNs. We sample event count at regular intervals to keep the notion of time consistent and allow NNs to learn temporal correlations between events at each pixel. We later show in the following section that sampling event counts at $10 \times$ the rate of the ground truth (10× temporally dense flow estimation than existing approaches) offers a good balance between the training time and flow prediction accuracy.

3.2 Sequential Training for Temporally Dense Optical Flow Estimation from an Event Stream

Estimating optical flow from an event stream challenges a basic assumption used during training of LSTMs and SNNs for sequential tasks; i.e. sequential inputs are of finite length. As an event stream is a contiguous long input, training NNs to estimate temporally dense optical flow is not trivial problem. First, NNs must be able to estimate optical flow without a network reset as a reset implies a loss of information from past events. Hence, to achieve temporally dense optical flow estimation, NNs must continuously process event counts and previous information without a network reset. Additionally, there are limited data augmentations that can be performed when we traverse an event stream and train during each epoch. This is because data augmentations must be uniform throughout any sequence. To address this particular issue, we can naively
split a long event stream into multiple smaller sequences consisting of 10 event counts as illustrated in Fig. 1(b) and train the LSTM using traditional sequential training methodologies. However, our experiments show that the network trained in this manner predicts optical flow with an unacceptably large average error, even with data augmentations. We observe that the prediction error starts increasing drastically after the first acceptable optical flow estimation. This is because traditional sequential training methodologies do not properly train NNs to ignore information from older events and focus on more recent events.

To prepare NNs for continuous operation on an event stream, we propose a two-step approach. Suppose that optical flow ground truth is available at every $m$ event counts. The first step is to create input sequences from an event stream consisting of $m \cdot n$ consecutive event counts (see Fig. 4(a)). We make sure that $m \cdot n$ is sufficiently large so that all important event counts are included for optical flow estimation. Doing so allows us to apply different data augmentations to each sequence and increases the number of data points for training. To make NNs aware of the previous event counts, we increase the length of each sequence by including $l$ additional event counts in front of the $m \cdot n$ event counts. The next step is to train NNs using the BPTT algorithm, but propagate gradient backward only to event counts that are within a window of interest equal to $m \cdot n$ (see Fig. 4(b)). Information from $l$ event counts beyond this window of interest is automatically treated as noise during training. Thus, we guarantee NNs to learn temporal correlations from $m \cdot n$ event counts by propagating gradient back in time.
3.3 Network Architectures for Temporally Dense Optical Flow Estimation

Encoder-decoder network architecture has been widely adapted by the prior works for optical flow estimation [14][17]. This architecture has multiple downsampling convolutional layers followed by upsampling convolutional layers. The first section of the network aims to encode spatio-temporal input into intermediate representations while the latter part utilizes these representations to estimate optical flow. We follow the same convention and construct an NN for temporally dense optical flow prediction by stacking LSTM cells/spiking neurons into multi-layer networks. We design each layer output to be of the same size as the popular EV-FlowNet outlined in [11]. All fully-connected operations of the LSTM cells/spiking neurons (see Fig. 3) are replaced with convolutional operations to handle image-like inputs, hence allowing the NN to draw spatial correlations among nearby events. Temporal correlations of events are then retained through the internal dynamics of the LSTM cells/spiking neurons. In the next section, we compare the performance of EV-FlowNet with LSTM network and SNN in terms of optical flow estimation accuracy.

4 Experimental Setup and Results

4.1 Dataset, Training and Evaluation Procedure

We demonstrate the effectiveness of the proposed temporally dense optical flow estimation on the DSEC dataset [28] which contains both high-resolution events and optical flow ground truths from daytime and nighttime outdoor driving under various lighting conditions. The events are recorded using a stereo event-based camera. Optical flow ground truths are derived from odometry ground truths and are publicly available for 18 scenarios. We split events and optical flow ground truths in each recording into a training and testing set using an 80/20 ratio. In other words, we pick the first 80% of the events and the corresponding ground truths in each recording to be a training set while we use the rest as a testing set. For training, only events from the left camera (after applying optical correction) are used for tabulating event counts. We randomly augment the events and optical flow ground truths by flipping them along vertical and horizontal directions and cropping them down to a size of $288 \times 384$. We train
Table 1: Comparison between various network architectures.

| Network architecture                          | Pred rate | AEE  | # Params |
|-----------------------------------------------|-----------|------|----------|
| EV-FlowNet                                    | 10 Hz     | 1.71 | 13M      |
| LSTM w/ typical seq training                  | 100 Hz    | 5.67 | 14M      |
| LSTM w/ proposed seq training                 | 100 Hz    | 1.28 | 14M      |
| SNN w/ proposed seq training and 4-bit outputs| 100 Hz    | 2.15 | 9M       |

The baseline and proposed networks with Adam optimizer for 20 epochs with an initial learning rate of $1 \times 10^{-4}$ and a batch size of 16. Since the optical flow ground truths in DSEC dataset are available at 10 Hz, we only perform backward-propagation at the end of each generated sequence where the ground truth is available. We trained all proposed NNs with $L_2$ loss that minimizes the squared differences between optical flow estimation and ground truth. The loss function can be mathematically expressed as:

$$
L = \sum_M \sum_N \| (u, v)_{\text{prediction}} - (u, v)_{\text{gt}} \|_2
$$

where $M$ is the total number of ground truths in an event stream and $N$ is the number of active pixels in the ground truth. $(u, v)$ represents optical flow magnitude along $(x, y)$ directions. Unless stated otherwise, we sample event counts at $10 \times$ the rate of optical flow ground truths. Thus, the proposed NNs estimate optical flow at a rate of 100 Hz.

For evaluation, we center crop events from each recording and obtain event counts of size $288 \times 384$ for optical flow estimation. We sequentially feed event count to the proposed NNs one by one and obtain optical flow estimation. Even though the proposed NNs produce temporally dense optical flows at 100 Hz, we only evaluate the predicted flows with ground truths at 10 Hz to calculate the accuracy (whenever reliable ground truths are available). The test set consists of 1601 ground truths from 18 driving scenes. We report the average of end-point errors (AEE) which is the mean of the Euclidean distance between the predicted flow and the ground truth. We also compute the percentage of pixels that have predicted errors greater than $k$ number of pixels (denoted as $k$PE). As predicted optical flow is sparse, we limit the calculation of all metrics to pixels that have at least one event from the last 10 event counts. We also omit the computations on areas that have no valid ground truth (no odometry information available).

### 4.2 Optical Flow Estimation Rate and Accuracy

Table 1 shows a comparison between various network architectures in terms of the optical flow estimation rate, AEE, and the number of parameters. To demonstrate shortcomings of the existing optical flow approaches, we train EV-FlowNet to make a prediction from event counts. Since the optical flow ground
Table 2: Comparison b/w sequential networks for flow estimation

| Network architecture | AEE   | 2PE  | 3PE  | Params | MultOps |
|----------------------|-------|------|------|--------|---------|
| LSTM                 | 1.28  | 15%  | 6%   | 1.00   | 1.00    |
| SNN w/ 4-bit outputs | 2.15  | 37%  | 17%  | 0.68   | 0.69    |
| SNN w/ 3-bit outputs | 2.84  | 50%  | 29%  | 0.68   | 0.63    |
| SNN w/ 2-bit outputs | 4.48  | 73%  | 54%  | 0.68   | 0.52    |

truths are available at 10 Hz, we split the event stream at the same interval and use each sequence as input to EV-FlowNet. Hence, EV-FlowNet expects inputs at a rate of 10 Hz for both inference and training. Thus, the prediction rate for EV-FlowNet is an order of magnitude lower than NNs trained in a sequential manner (see column 2).

NNs estimate optical flows poorly when traditional sequential training methods are used. We show that LSTM networks trained with such traditional methods perform poorly in terms of prediction accuracy (see row 3). LSTM networks trained through our proposed sequential method (with \( n=2 \) and \( l=10 \)), however, are able to predict temporally dense optical flows while achieving the lowest AEE on the DSEC dataset. We recorded an AEE of 19.7\%, lower than that of EV-FlowNet, owing to the ability to draw longer correlations back in time through sequential learning. Our visual analysis indicates that LSTM network outperforms EV-FlowNet in a scenario with only few reliable events such as events generated from a tree line under low illumination (see Fig. 5).

We also show that SNN can estimate temporally dense optical flow similar to the LSTM network but with fewer parameters (see the last row of Table 1). This comes with a slightly higher AEE. We consider it acceptable as the AEE is still lower than the average flow magnitude of 7.73 pixels in the test set. Note that the magnitude of the flow ground truths exceed 20 pixels around the 80th percentile.

4.3 Advantages and Disadvantages of Computing with SNN

The benefits and downsides of using SNN based networks can be seen through the experimental results reported in Table 2. The number of parameters and multiply operations reported here are normalized by those values of the LSTM network. In addition to the advantage of SNN’s high fidelity with event streams, two other important benefits can be identified. The first is a significant reduction in the number of parameters required as shown in Table 2. This is because SNNs use simpler dynamics than LSTM networks to capture temporal information. Another benefit is the computational savings resulting from input sparsity. Zero inputs ideally incur little or no energy consumption if the SNN is implemented on event-driven hardware. To quantify the energy savings, we track the percentage of non-zero inputs received by spiking neurons in each layer and find the total number of arithmetic operations. Because multiplication is an operation
Table 3: Effect of event count sampling rate on LSTM network.

| Flow prediction rate | AEE | 1PE | 2PE | 3PE | 4PE | 5PE |
|----------------------|-----|-----|-----|-----|-----|-----|
| 50 Hz                | 2.30| 70% | 33% | 19% | 12% | 8%  |
| 100 Hz               | 1.28| 47% | 15% | 6%  | 3%  | 2%  |
| 150 Hz               | 1.38| 52% | 17% | 7%  | 4%  | 2%  |

With significant energy overhead, we use the total number of multiplying operations as a proxy to evaluate power savings. The last column of Table 3 shows how the number of multiplying operations reduces with the number of bits the SNN uses for communication between layers. This is a result of the threshold function that generates an output only if the membrane potential exceeds the threshold values. When the number of bits decreases, the number of threshold levels also reduces, allowing only dominant activities to pass through, leading to outputs full of zeros. Computational savings are achieved when sparse outputs are used as inputs by the following spiking neuron layer, thus reducing the number of multiplying operations. However, sparse inputs negatively affect flow prediction error, creating a trade-off between computational savings and prediction accuracy. Depending on the target application, one may utilize a metric like kPE which gives the percentage of error beyond k pixels to select a NN with desired energy efficiency and prediction accuracy.

4.4 Effect of Input Rate on the Prediction Accuracy

Because the proposed NNs estimate optical flow at the same rate as the input (every event count), we measure the trade-offs of estimating optical flow at different input rates. For time-critical applications, optical flow estimation at a high rate is desirable. However, this entails more computations at both training and deployment. Accumulating event counts for a long duration reduces the number of computations, but it risks losing temporal information. On the DSEC dataset, we find that increasing the event count sampling rate beyond 100 Hz does not lead to significant improvement in flow prediction error (see Table 3). In fact, AEE is slightly higher for 150 Hz than 100 Hz, but both can be considered to have similar accuracy as all kPE values are relatively close to each other.

5 Conclusion

In this work, we cast the optical flow estimation problem as a sequential learning task to achieve temporally dense optical flow estimation. We show that traditional sequential training approaches are not suitable for training sequential networks for flow estimation from a continuous event stream. We propose modifications to the training methodology that enable the network to ignore information from older events and to focus more on recent events for optical
flow estimation. This enables a 10× temporally dense flow estimation over existing flow estimation approaches. Results on LSTM networks indicate a 19.7% accuracy improvement over EV-FlowNet owing to the ability to draw longer temporal correlations from an event stream. We also show that SNNs can be used to achieve temporally dense optical flow estimations with higher (compared to LSTMs) but acceptable prediction error. Given that SNNs handle event-driven operations more efficiently than LSTMs, this work validates the potential for energy-efficient platforms where an event-based camera and a neural network can work together to produce fast and temporally dense optical flows.

Acknowledgements

This work was supported in part by, Center for Brain-inspired Computing (C-BRIC), a DARPA sponsored JUMP center, Semiconductor Research Corporation (SRC), National Science Foundation, the DoD Vannevar Bush Fellowship, and IARPA MicroE4AI.

References

[1] R. Quan, L. Zhu, Y. Wu, and Y. Yang, “Holistic LSTM for pedestrian trajectory prediction,” IEEE transactions on image processing, vol. 30, pp. 3229–3239, 2021.

[2] J. R. Serres and F. Ruffier, “Optic flow-based collision-free strategies: From insects to robots,” Arthropod structure & development, vol. 46, no. 5, pp. 703–717, 2017.

[3] J. Janai, F. Güney, A. Behl, A. Geiger, et al., “Computer vision for autonomous vehicles: Problems, datasets and state of the art,” Foundations and Trends® in Computer Graphics and Vision, vol. 12, no. 1–3, pp. 1–308, 2020.

[4] A. Singh, Optic flow computation: a unified perspective. IEEE computer society press Los Alamitos, 1991, vol. 3.

[5] S. S. Beauchemin and J. L. Barron, “The computation of optical flow,” ACM computing surveys (CSUR), vol. 27, no. 3, pp. 433–466, 1995.

[6] B. K. Horn and B. G. Schunck, “Determining optical flow,” Artificial intelligence, vol. 17, no. 1-3, pp. 185–203, 1981.

[7] A. Dosovitskiy, P. Fischer, E. Ilg, P. Hausser, C. Hazirbas, V. Golkov, P. Van Der Smagt, D. Cremers, and T. Brox, “FlowNet: Learning optical flow with convolutional networks,” in Proceedings of the IEEE international conference on computer vision, 2015, pp. 2758–2766.
[8] A. Ranjan and M. J. Black, “Optical flow estimation using a spatial pyramid network,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 4161–4170.

[9] D. Sun, X. Yang, M.-Y. Liu, and J. Kautz, “PWC-net: CNNs for optical flow using pyramid, warping, and cost volume,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 8934–8943.

[10] G. Gallego, T. Delbrück, G. Orchard, C. Bartolozzi, B. Taba, A. Censi, S. Leutenegger, A. J. Davison, J. Conradt, K. Daniilidis, et al., “Event-based vision: A survey,” *IEEE transactions on pattern analysis and machine intelligence*, vol. 44, no. 1, pp. 154–180, 2020.

[11] A. Z. Zhu and L. Yuan, “EV-FlowNet: Self-supervised optical flow estimation for event-based cameras,” in *Robotics: Science and Systems*, 2018.

[12] C. Ye, A. Mitrokhin, C. Fermüller, J. A. Yorke, and Y. Aloimonos, “Unsupervised learning of dense optical flow, depth and egomotion with event-based sensors,” in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2020, pp. 5831–5838.

[13] Z. Li, J. Shen, and R. Liu, “A lightweight network to learn optical flow from event data,” in *2020 25th International Conference on Pattern Recognition (ICPR)*. IEEE, 2021, pp. 1–7.

[14] A. Z. Zhu, L. Yuan, K. Chaney, and K. Daniilidis, “Unsupervised event-based learning of optical flow, depth, and egomotion,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 989–997.

[15] C. Lee, A. K. Kosta, A. Z. Zhu, K. Chaney, K. Daniilidis, and K. Roy, “Spike-FlowNet: event-based optical flow estimation with energy-efficient hybrid neural networks,” in *European Conference on Computer Vision*. Springer, 2020, pp. 366–382.

[16] C. Lee, A. K. Kosta, and K. Roy, “Fusion-FlowNet: Energy-efficient optical flow estimation using sensor fusion and deep fused spiking-analog network architectures,” in *2022 International Conference on Robotics and Automation (ICRA)*. IEEE, 2022, pp. 6504–6510.

[17] J. Hagenaars, F. Paredes-Vallès, and G. De Croon, “Self-supervised learning of event-based optical flow with spiking neural networks,” *Advances in Neural Information Processing Systems*, vol. 34, pp. 7167–7179, 2021.

[18] N. J. Sanket, C. M. Parameshwara, C. D. Singh, A. V. Kuruttukulam, C. Fermüller, D. Scaramuzza, and Y. Aloimonos, “EvDodgeNet: Dynamic obstacle dodging with event cameras,” in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 10 651–10 657.
[19] S. Hochreiter and J. Schmidhuber, “Long short-term memory,” *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[20] A. Graves, A.-r. Mohamed, and G. Hinton, “Speech recognition with deep recurrent neural networks,” in *2013 IEEE international conference on acoustics, speech and signal processing*. IEEE, 2013, pp. 6645–6649.

[21] I. Sutskever, O. Vinyals, and Q. V. Le, “Sequence to sequence learning with neural networks,” *Advances in neural information processing systems*, vol. 27, 2014.

[22] M. Davies, N. Srinivasa, T.-H. Lin, G. Chinya, Y. Cao, S. H. Choday, G. Dimou, P. Joshi, N. Imam, S. Jain, et al., “Loihi: A neuromorphic manycore processor with on-chip learning,” *IEEE Micro*, vol. 38, no. 1, pp. 82–99, 2018.

[23] M. V. DeBole, B. Taba, A. Amir, F. Akopyan, A. Andreopoulos, W. P. Risk, J. Kusnitz, C. O. Otero, T. K. Nayak, R. Appuswamy, et al., “TrueNorth: Accelerating from zero to 64 million neurons in 10 years,” *Computer*, vol. 52, no. 5, pp. 20–29, 2019.

[24] G. Orchard, E. P. Frazy, D. B. D. Rubin, S. Sanborn, S. B. Shrestha, F. T. Sommer, and M. Davies, “Efficient neuromorphic signal processing with loihi 2,” in *2021 IEEE Workshop on Signal Processing Systems (SiPS)*. IEEE, 2021, pp. 254–259.

[25] W. Gerstner, W. M. Kistler, R. Naud, and L. Paninski, *Neuronal dynamics: From single neurons to networks and models of cognition*. Cambridge University Press, 2014.

[26] W. Ponghiran and K. Roy, “Spiking neural networks with improved inherent recurrence dynamics for sequential learning,” in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 36, no. 7, 2022, pp. 8001–8008.

[27] M. Courbariaux, I. Hubara, D. Soudry, R. El-Yaniv, and Y. Bengio, “Binarized neural networks: Training deep neural networks with weights and activations constrained to+ 1 or-1,” *arXiv preprint arXiv:1602.02830*, 2016.

[28] M. Gehrig, W. Aarents, D. Gehrig, and D. Scaramuzza, “DSEC: A stereo event camera dataset for driving scenarios,” *IEEE Robotics and Automation Letters*, vol. 6, no. 3, pp. 4947–4954, 2021.