Discrete time convolution for fast event-based stereo

Kaixuan Zhang[1,3*] Kaiwei Che[2,3] Jianguo Zhang[1,4]
Jie Cheng[3] Ziyang Zhang[3] Qinghai Guo[3] Luziwei Leng[3*†]
[1] Department of Computer Science and Engineering, Southern University of Science and Technology, China
[2] Department of Electrical and Electronic Engineering, Southern University of Science and Technology, China
[3] ACS Lab, Huawei Technologies, Shenzhen, China [4] Peng Cheng Lab, Shenzhen, China

Abstract

Inspired by biological retina, dynamical vision sensor transmits events of instantaneous changes of pixel intensity, giving it a series of advantages over traditional frame-based camera, such as high dynamical range, high temporal resolution and low power consumption. However, extracting information from highly asynchronous event data is a challenging task. Inspired by continuous dynamics of biological neuron models, we propose a novel encoding method for sparse events - continuous time convolution (CTC) - which learns to model the spatial feature of the data with intrinsic dynamics. Adopting channel-wise parameterization, temporal dynamics of the model is synchronized on the same feature map and diverges across different ones, enabling it to embed data in a variety of temporal scales. Abstracted from CTC, we further develop discrete time convolution (DTC) which accelerates the process with lower computational cost. We apply these methods to event-based multi-view stereo matching where they surpass state-of-the-art methods on benchmark criteria of the MVSEC dataset. Spatially sparse event data often leads to inaccurate estimation of edges and local contours. To address this problem, we propose a dual-path architecture in which the feature map is complemented by underlying edge information from original events extracted with spatially-adaptive denormalization. We demonstrate the superiority of our model in terms of speed (up to 110 FPS), accuracy and robustness, showing a great potential for real-time fast depth estimation. Finally, we perform experiments on the recent DSEC dataset to demonstrate the general usage of our model.

1. Introduction

Depth estimation has been viewed as one of the key problems in computer vision, with a variety of applications such as robotics, autonomous driving, augmented reality and medical imaging. Multi-view stereo matching solves this problem by reconstructing 3D scenes based on pixel differences of the same physical point from images taken from multiple views. Current deep learning models for stereo matching mainly work with static images produced by frame-based cameras. These sensors have drawbacks such as high power consumption, low dynamical range and low data rate, which limit their usage in edge computing platforms or high-speed scenarios. Inspired by biological retina, the recently emerging dynamical vision sensor, or event-based sensor, addresses these problems by transmitting events of instantaneous changes of pixel intensity, in millisecond-level temporal resolution. When the change of brightness at a pixel exceeds a certain threshold, the sensor will fire an event, which is a four-dimensional vector including spatial coordinates, polarity and timestamp. Since the event-based sensor transmits events rather than frames, motion blur caused by high-speed movements of objects during exposure can be avoided. This property makes it perfectly suitable for high-speed vision tasks, such as object detection, tracking and obstacle avoidance, etc.

1.1. Event-based stereo matching

Most of the works in stereo matching based on deep learning are established on image datasets [20, 32, 52, 56, 59]. Therefore, to take advantage of accumulated knowledge in this field, a straightforward approach is to convert the highly asynchronous event sequences into frame-based images. Following this idea, various methods have been proposed. The so-called handcrafted methods [25, 31, 35, 36, 40, 53, 58, 64] directly convert events to event frames based on the four dimensional information of each event. For example, in [36, 53], stacking based on time (SBT) merges events into predefined temporally neighboring bins and sums up polarity values of events if they share the same spatial coordinate; stacking based on the number of events

*These authors contribute equally to this work.
†Corresponding author. lengluziwei@huawei.com

(SBN) creates a frame by accumulating a fixed number of events in the history and preserves the polarity of the last event if they overlap. [25, 31] stored histograms of events of different polarities in different channels to avoid information loss due to polarity cancellation. [64] preserved the timestamp of the last positive and negative events at every location. [65] created voxel grids by interpolation based on timestamps of events. [34, 47] applied specially designed asynchronous convolution for sparse events data. Other methods use additional information for stereo matching. TSES [62] utilized the velocity of the camera to approximate optical flow and build time synchronized event disparity volumes. Semi-Dense 3D [61] used camera pose information to integrate observations over time to produce semi-dense depth maps. Some end-to-end learning methods of events encoding have also been proposed to tackle specific tasks. [15] used events measurement field to convert events into grid-based representation. [9] used time-surface with linear time decay to construct event images. However, predicting dense disparities from sparse events is still challenging. The event queue method proposed in [51] used a continuous fully-connected layer to learn a list of weights for events of different timestamps. The work adopted a previous architecture for image-based stereo matching [52] and was the first to predict dense disparity images based on events. However, its accuracy on local structures or edges is still insufficient. Recent works addressed this problem by utilizing intensity images. Taking [51] as a backbone, [2] enhanced the quality of predicted disparity on local structures by training an image-reconstruction sub-network with semantic attention, which complements the event feature by spatially-adaptive denormalization (SPADE) [42, 48]. Using SBN for events encoding, [37] combined events and intensity images in a sequential manner and correlates them to estimate dense depth value. Although state-of-the-art performances have been achieved, the high computational cost and memory consumption make these models quite expensive to deploy in practice, especially for high-speed scenarios when intensity images are prone to blur.

1.2. Recurrent neural networks

An alternative encoding method for events is to use recurrent neural networks (RNNs), given their inherent ability to encode temporal sequences. However, fully-connected RNNs are not efficient for information extraction of images. A natural thought is to incorporate RNNs into convolutional operations. Recurrent convolutional neural network (RCNN) [29] takes input from the last layer and combines it with recurrent input from the current layer. By using intra-layer recurrent connections, RCNN can integrate context information. Different embodiments of RCNN include convolutional long short-term memory (ConvLSTM) [55] and convolutional gated recurrent units (ConvGRU) [4], where additional gating variables were used for memory modulation. [39] applied a modified version of LSTM [19] to event-based recognition, but the model was not specifically designed to preserve spatial information.

Different from traditional RNNs constructed with artificial neurons, spiking neural networks (SNNs) [30] uses spiking neuron models inspired by biology with inherent self-recurrence. The neuron evolves its membrane potential with an individual time constant and resets to a reset potential whenever it spikes. There has been an increasing number of applications of SNNs in deep learning [5, 12, 21, 22, 28, 38, 45, 50, 54, 60], and the network's asynchronous nature makes it an ideal solution for event-based tasks [7, 8, 23, 27, 41, 57]. However, the training of SNNs is challenging due to discontinuous spikes which is incompatible with gradient-based back-propagation algorithms [38]. The liquid time-constant network (LTC) [18, 26], an expansion of the continuous time RNN [13], circumvents this problem by using continuous valued activation functions for its neuron, whose dynamics is modulated by an input-dependent system time constant. However, the LTC was only applied for low dimensional temporal sequence modeling and it lacks the ability to encode high dimensional spatial features.

1.3. Contribution

Inspired by convolutional RNNs [4, 29, 55] and continuous temporal dynamics of biological neuron models [18, 30], we propose a novel event processing method combining merits of both sides. Motivated by recent studies [42, 48], we enhance the quality of predicted disparity with SPADE and develop an efficient framework for event-based stereo matching. In summary, contributions of this paper are four-folds:

1. We develop continuous time convolution (CTC), an expansion of LTC, for encoding high dimensional spatial-temporal data. The model adopts channel-wise parameterization which enables feature maps to embed the data in a variety of temporal scales through end-to-end training. Based on CTC, we further propose an abstracted model, discrete time convolution (DTC), enabling faster evolution and stable training.
2. We demonstrate the advantage of CTC and DTC over other event encoding methods on a set of criteria of event-based stereo matching, on the Multi Vehicle Stereo Event Camera (MVSEC) [63] dataset.
3. We further develop a dual-path architecture where underlying edge information from original event frames is extracted to improve local contours of estimated disparity. Through streaming experiments we demonstrate the superiority of our model in terms of speed (up to 110 FPS), accuracy and robustness.

4. Finally, we perform preliminary experiments on DSEC [16], a recent large-scale outdoor event stereo dataset, demonstrating the general usage of our model.

2. Method

An event is represented as a four-dimensional vector, (x, y, p, t), where (x, y) denote the spatial coordinate of the event, p denotes the polarity indicating the directional change of the pixel intensity in log scale and t denotes the time when the event happens. If the brightness change of the pixel exceeds the threshold, then $p = 1$, otherwise -1.

2.1. Event representation

How to convert events into event frames is essential to downstream tasks. In [2, 51], an event queue was created for each pixel using SBN with the timestamp information of each event additionally preserved for the training of a continuous fully connected layer. By accumulating a fixed number of events into a plane, the event queue method ensures the abundance of spatial information, but it loses the information of those events that exceed the capacity of the queue. SBT [36] merges events into frames by a fixed time interval. When the number of events is sufficient, it could maintain certain temporal information while keeping dense spatial information. Suppose the duration of an event stream is Δt and all events are compressed into n frames. The value of each pixel in frame f is the accumulated polarity of events:

$$P(x, y) = \text{sign}(\sum_{t \in T} p(x, y, t)) \tag{1}$$

where P is the value of the pixel at (x, y), t is the timestamp, p is the polarity of the event and $T \in [\frac{(f-1)\Delta t}{n}, \frac{f\Delta t}{n}]$. The sign operator projects the accumulated polarities into $(-1, 0, 1)$, which allows the event frame to be more robust in case the left or right camera generates more events at each pixel than the other. It also decreases data storage space for potential hardware applications, similar approaches were taken in [62]. The advantages of this method are two-folds. First, it's robust to certain noise caused by environment. Second, because it keeps firing frames at a fixed rate, it maintains stable temporal information to a large extent. [36] mentioned that if there are too few events happening during the interval, SBT may produce a very sparse event image. This intrinsic limitation can be alleviated in our proposed method, as we demonstrate in following sections.

2.2. LTC network

The LTC network [18] is an expansion of continuous-time RNN (CT-RNN) [13], which can be described by an ordinary differential equation (ODE):

$$\frac{dx(t)}{dt} = -\frac{x(t)}{\tau} + f(x(t), I(t), t, \theta) \tag{2}$$

where τ characterizes the speed and the coupling sensitivity of the dynamical system, $x(t)$ is the hidden state, $I(t)$ is the input, t represents time and f is a neural network parameterized by θ. It has been proved that CT-RNN can approximate any finite time trajectory of a n-dimensional dynamical system, by embedding the system into a higher dimensional one which defines an RNN [13]. LTC further enhances its ability by integrating f into the time constant of the system:

$$\begin{aligned}\frac{dx(t)}{dt} = -\left[\frac{1}{\tau} + f(x(t), I(t), t, \theta)\right] x(t) + \\ f(x(t), I(t), t, \theta) A\end{aligned} \tag{3}$$

where the system time constant becomes an input-dependent term $\frac{\tau}{1+\tau f(x(t),I(t),t,\theta)}$ and A is a scale parameter. The ODE realizes a system of stiff equations [46], and can be solved by fusing the explicit and implicit Euler method. This equation can be loosely related to the computational models of neural dynamics in small species, when written in the form:

$$\begin{aligned}\frac{dx(t)}{dt} = -\left[\frac{1}{\tau_{\text{m}}} + \frac{W g(t)}{C_{\text{m}}}\right] x(t) \\ + \frac{W g(t)}{C_{\text{m}}} E_{\text{rev}} + \frac{E_{\text{leak}}}{\tau_{\text{m}}}\end{aligned} \tag{4}$$

where f in Eq. 3 is now defined by $\frac{W g(t)}{C_{\text{m}}}$, $x(t)$ now represents the membrane potential of the neuron at time t, τ_{m} is the membrane time constant, W is the input synaptic strength, $g(t)$ is the synaptic input, E_{leak} is the resting potential and E_{rev} is the reversal synaptic potential. The model can be viewed as a non-spiking form of the leaky integrate and fire (LIF) neuron with conductance synapse [44], without specifically defining its synaptic dynamics. When neglecting the influence of the reversal synaptic potential, Eq. 4 becomes:

$$\frac{dx(t)}{dt} = \frac{E_{\text{leak}} - x(t)}{\tau_{\text{m}}} + \frac{W g(t)}{C_{\text{m}}} \tag{5}$$

where the system time constant is untangled from its input and the model becomes a traditional CT-RNN. The LTC network was only applied for low dimensional temporal sequence modeling. To encode high-dimensional spatial-temporal data, we extend the fully connected $W g(t)$ to have a convolution structure.

2.3. Continuous time convolution

In the fully connected structure, the synaptic input of an LTC neuron contains inputs from all the other neurons. A straight copy to the convolution structure will generate numerous parameters which scale with the channel depth. In addition, when encoding highly sparse event frames, it is often the case that there are few or even no signals in neighboring areas, leading to little context information. Therefore, we only preserve connections from the previous layer

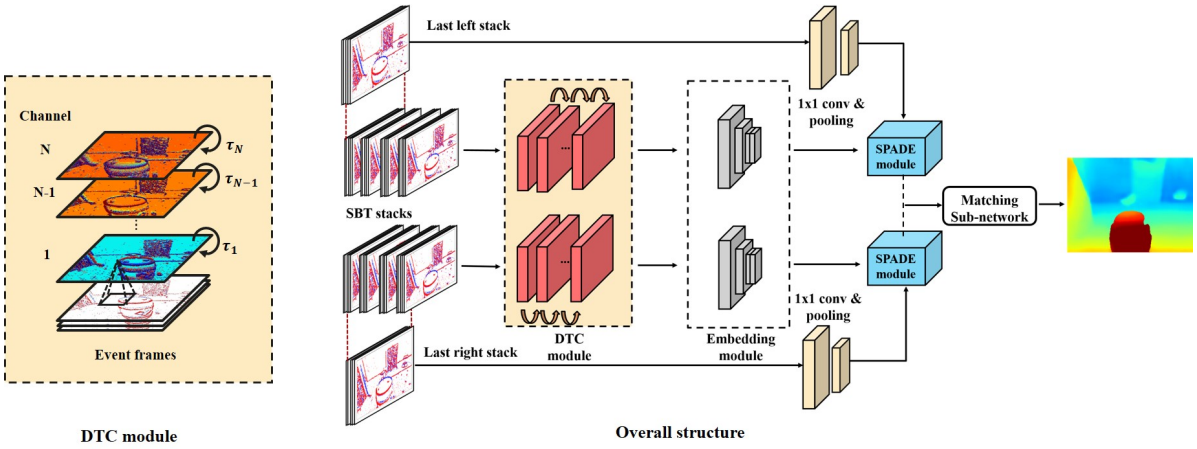

Figure 1. The left part is the DTC module, the right part is the overall structure of DTC-SPADE. The DTC module takes as input stacks of event frames prepared by SBT. During this process, channels are trained to accumulate information at different temporal scales characterized by different values of τ. The following embedding module further extracts spatial information from the DTC module. Meanwhile, 1×1 convolution and pooling operations are applied on the last event stack to fuse information across channels and shrink the spatial size. These auxiliary feature maps are then used to modulate feature maps from the embedding module in the SPADE module. Outputs from the left and right pathways are then fed to the matching sub-network to generate disparity maps.

and leave out intra-layer recurrent connections. In addition, we adopt channel-wise parameterization instead of pixel-wise to constrain each feature map with synchronized dynamics, which further decreases the total number of parameters. The resulting convolution LTC neuron and its simplified version can be formulated as:

$$\frac{dx_{cij}(t)}{dt} = -\left[\frac{1}{\tau_{\mathrm{m},c}} + \frac{I_{cij}(t)}{C_{\mathrm{m},c}}\right] x_{cij}(t) + \frac{I_{cij}(t)}{C_{\mathrm{m},c}} E_{\mathrm{rev},c} + \frac{E_{\mathrm{leak},c}}{\tau_{\mathrm{m},c}} \tag{6}$$

$$\frac{dx_{cij}(t)}{dt} = \frac{E_{\mathrm{leak},c} - x_{cij}(t)}{\tau_{\mathrm{m},c}} + \frac{I_{cij}(t)}{C_{\mathrm{m},c}} \tag{7}$$

$$I_{cij}(t) = \sum_h \sum_k w_{chk} P^t_{h+i,k+j} \tag{8}$$

with $Wg(t)$ in the previous section specified by $I_{cij}(t)$, which represents the convolution input on channel c at location i, j from the event frame pre-processed by SBT, h and k are spatial coordinates on the input plane. The output of the LTC neuron is normalized by a parametrical sigmoid function $\sigma(x_{cij}) = 1/(1 + \exp(\gamma_c(\mu_c - x_{cij})))$, where γ_c and μ_c are trainable parameters that scale and shift x_{cij}. We term both convolution LTC (convLTC) and convolution LTC without reversal potential (convLTCOR) as continuous time convolution (CTC). For simulation, we solve the dynamical equations numerically by fusing the implicit and explicit Euler methods (see the supplement material for derivation).

2.4. Discrete time convolution

A precise approximation of CTC requires numerically integrating multiple small steps [17, 18]. In the original work, the LTC neuron was evolved at a frequency six times higher than the input sampling rate, leading to a six times slower output rate for an equal temporal span of the input. However, changing to a larger numerical step to accelerate the output rate could potentially cause unstable results. A recent study [49] proved that, when certain stability conditions are satisfied, continuous-time recurrent neural networks can be approximated by discrete time recurrent neural networks. The dynamics of the convLTCOR model is mainly characterized by its membrane time constant τ_{m}, abstracted from this intuition, we develop the discrete time convolution model (DTC), formulated as:

$$x^t_{cij} = \sigma(\tau_c x^{t-1}_{cij} + I_{cij}(t)) \tag{9}$$

where $I_{cij}(t)$ is defined the same as in Eq. 8, x^t_{cij} represents the pixel value on channel c at location i, j of the corresponding feature map. The time constant τ_c is assigned channel-wisely as in CTC, which controls the aggregation strength of the neuron's previous state and σ is the sigmoid function. A conceptual plot of DTC is depicted in Fig. 1.

2.5. Network architecture

Successful current works for image-based stereo matching generally perform a four-step pipeline: feature embedding, matching volume, regularization and refinement. State-of-the-art works for event-based stereo matching [2,

37, 51] followed this convention. Our framework uses [51] as a backbone, with major modifications in the feature embedding sub-network (Fig. 1).

Event sequences are first converted by SBT to form event stacks each containing multiple frames. They are then fed to a spatial-temporal encoding module constructed with DTC or CTC layers. The output is then fed to the spatial embedding module, followed by the matching and regularization module as in [51]. We term this architecture as CTC-PDS or DTC-PDS respectively, depending on the neuron type in the spatial-temporal encoding module.

However, events are often highly sparse, directly applying a stack of convolution operations on them could cause the loss of semantic information and might be not sufficient to reconstruct edges or details of local structures on disparity maps. Inspired by recent studies [2, 6, 42, 48], we further develop a dual-path structure for feature embedding fused by SPADE with multi-scale dilated convolution. As Fig. 1 depicts, the first path encodes spatial-temporal information by CTC or DTC, followed by the spatial embedding module, which is identical to the CTC/DTC-PDS network. The second path extracts underlying edge information from the last stack of the original event frames (align with the ground truth disparity in time). We use 1×1 convolution layers to fuse the information across channels and average pooling to adjust the spatial dimension. Conditioned on this auxiliary feature map, the SPADE module extracts modulation parameters which shift and scale the feature map of the spatial embedding module. Its output is formulated as

$$\hat{h}_{b,c,y,x} = \gamma_{c,y,x}(\mathbf{s}) \frac{h_{b,c,y,x} - \mu_c}{\sigma_c} + \beta_{c,y,x}(\mathbf{s}) \tag{10}$$

where b, c, y, x denote batch index, channel index and spatial coordinates, $\mathbf{s}$ is the last SBT stack, $\gamma_{c,y,x}(\mathbf{s})$ and $\beta_{c,y,x}(\mathbf{s})$ are modulated parameters trained on $\mathbf{s}$, $h_{b,c,y,x}$ is the activation from the spatial embedding module. μ_c and σ_c are mean and standard deviation of batch normalization.

We use the same matching sub-network and loss function as [51]. Left features output from SPADE are concatenated with corresponding shifted right features and matched by a stack of convolution operations. After the matching operation, we get a 4D matching volume of size $\frac{c}{8} * \frac{d}{4} * \frac{h}{4} * \frac{w}{4}$, where d denotes the disparity dimension. Then, the regularization module uses an hourglass structure to mix information across disparities and channels and obtains a cost volume of size $\frac{d}{2} * h * w$. More details about the network architecture can be found in the supplementary material. We use sub-pixel cross-entropy loss to train the model:

$$\mathbf{L}(\Theta) = \frac{1}{wh} \sum_{y,x} \sum_{j} \text{Laplace}(d(j) | \mu = D^{GT}_{y,x}, b) * \log(\underset{j}{\text{softmin}}(C_{j,y,x})) \tag{11}$$

Method	MDE, [cm] ↓	1PA, [%] ↑	No. param.
Hand-crafted	17.9±0.6	88.1±0.4	0
ConvGRU	55.9±28.5	45.9±21.9	28800
ConvLSTM	20.6±1.9	82.2±4.5	38272
Event queue	16.9±1.0	89.3±1.4	12672
DTC	15.4±0.1	91.2±0.1	**1632**
CTC	**15.1**±0.3	**91.2**±0.4	1760

Table 1. Results of different event encoding methods on split one.

where w, h denote the width and height of disparity map, j denotes the disparity index of the cost volume. We set diversity $b = 2$ and the disparity of ground truth in position (x, y) as mean, like [51, 52] do. So $d(j) = 2 * j$ denotes the number of disparity. Finally, we use sub-pixel estimator [52] to produce a disparity map:

$$\widehat{D}_{y,x} = \sum_{j} d(j) * \underset{j:|\hat{j}-j| \leq \delta}{\text{softmin}} (C_{j,y,x}) \tag{12}$$

with $\hat{j} = \arg\min_{j}(C_{j,y,x})$, where δ denotes the window size, $C_{j,y,x}$ represents the cost in pixel (j, y, x).

3. Experiments

We conduct our experiments on the MVSEC dataset [63], which contains depth information collected by a LIDAR sensor and event streams obtained from two event cameras with corresponding 20 Hz intensity images at 346×260 resolution. We split and preprocess the Indoor Flying dataset from the MVSEC using the same setting as [2, 51, 62]. In split one, 3110 samples from the Indoor Flying 2 and 3 are used as the training set while 861 and 200 samples from the Indoor Flying 1 are used as the test set and validation set. In split three, 2600 samples from the Indoor Flying 1 and 2 are used as the training set while 1343 and 200 samples from the Indoor Flying 3 are used as the test set and validation set. We use the mean depth error (MDE), one-pixel-accuracy (1PA), median depth error and mean disparity error as evaluation metrics for the dense disparity ground truth. In addition, we perform preliminary experiments on the recent large-scale outdoor event stereo dataset, DSEC [16], to further demonstrate the general usage of our model. Our models are constructed with PyTorch. For training details see the supplement material.

3.1. Comparison of event encoding methods

In this subsection, we compare CTC and DTC with different event encoding approaches, including event queue method [51], convLSTM, convGRU and the hand-crafted method [64] mentioned in sec 1.1. We perform dense disparity estimation where all locations on the ground truth are

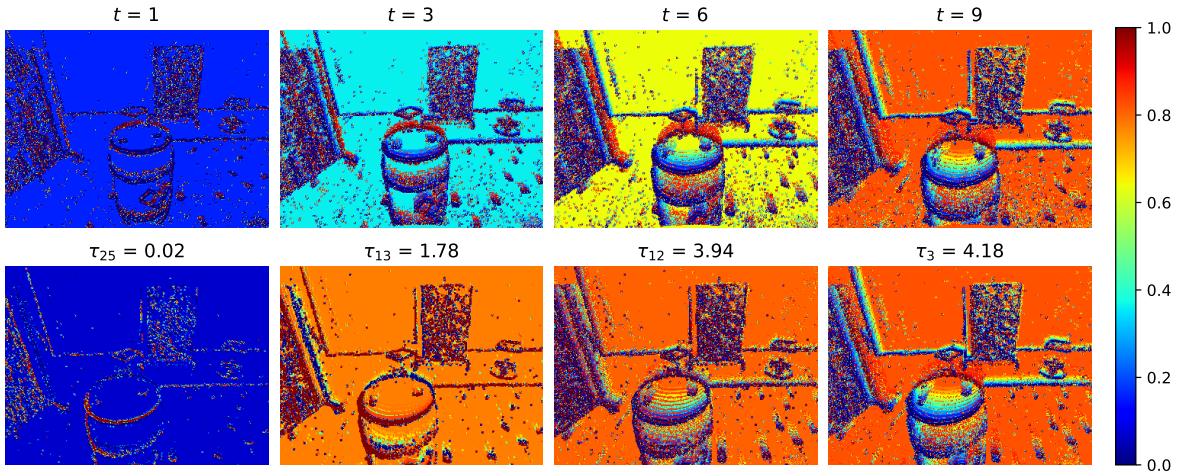

Figure 2. Feature maps of DTC. The upper row shows the evolution of a feature map at four different time steps ($t = 1, 3, 6, 9$). The feature map aggregates its past states and gradually forms a denser spatial representation. The bottom row shows four feature maps with different time constants at the end of evolution. Channels with larger τ remember more history than those with smaller τ.

used for evaluation, following [2,37,51]. For a fair comparison of all encoding methods, we use the same spatial embedding, matching and regularization module as [51]. The network was trained on split one for three times using different random seeds. For all methods except the event queue and hand-crafted methods, we use SBT to convert events into event frames. Specifically, $\Delta t = 50\text{ms}$ of events are compressed into a stack of $n = 5$ frames with each frame merging $T = 10\text{ms}$ of events, following Eq. 1. The input channel numbers are set to 5 for convLSTM, convGRU and our methods and the output channel number is set to 32. The parameters of SBT actually corresponds to the temporal resolution of the network, which is defined by $T = \frac{\Delta t}{n}$. For each training sample, the model reviews 15 preceding event stacks to accumulate temporal information. For every trial, we chose the checkpoint with the best 1PA on the validation set for testing. Tab. 1 shows the comparison on average 1PA, MDE and number of parameters. The results demonstrate that CTC and DTC have similar precision, and both outperform all the other methods in 1PA and MDE, with significantly fewer amount of parameters. Our methods also maintain certain robustness to a range of different SBT parameters (see supplement material). Note that for CTC, we use simulation results from the convLTCOR model. Empirically we found that the training of the convLTC model was unstable, during which gradients sometimes tended to vanish. This could be due to the convolutional input term in the denominator of the system time constant was not properly normalized; more studies are needed for this model.

Time constants of DTC

During the evolution, feature maps of DTC update their inputs meanwhile accumulating past states. After training, channels with larger τ remember more history than those with smaller τ, as shown in Fig. 2. Similar phenomenon is observed on the feature map of CTC (see supplement material). In the event queue method, a stack of 3D-convolution layers are trained to generate a set of temporal weight kernels which are multiplied to polarities stored in the queue according to their timestamps. These weight kernels can be understood as the importance of the network trained to assign over events at different moments. As shown in Fig. 3, DTC is trained to capture spatial features in a range of temporal scales, offering abundant causal and contextual information for downstream modules of the network. In con-

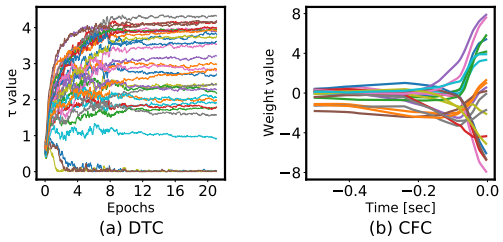

Figure 3. Time constants of DTC and kernel weights of event queue method. (a) shows the history of τ in the training process. We apply positive constraint on τ during training for a stable accumulation of past history. It can be seen that τ was trained to cover a wide span of values, enabling DTC to encode the data in a range of temporal scales. (b) shows a set of weight kernels of the event queue method after training. Their value distributions indicate that the network was trained to concentrate on more recent events.

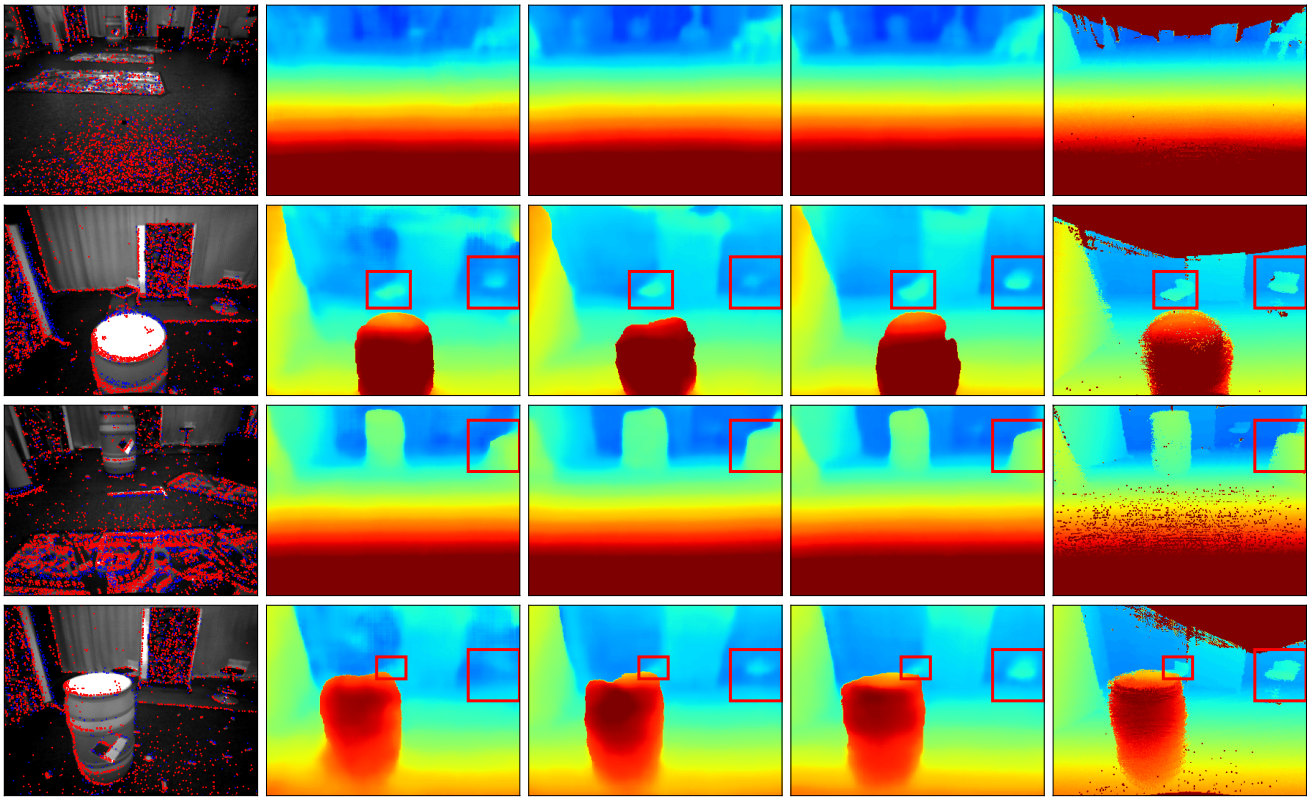

Figure 4. Qualitative comparison on the Indoor Flying dataset of MVSEC. The most left column are events overlapped on corresponding gray-scale images. Disparity maps of DDES, DTC-PDS, DTC-SPADE and the ground truth are on same frames, which are #100 from sequence 1, #340 from sequence 1, #1700 from sequence 3, #980 from sequence 1. We run the released code of DDES for dense disparity estimation and generate the baseline disparity map. Our proposed model could detect more spatial information than the baseline. More details about the plot, including the colormap setup are in the supplement material.

Method	EO	Mean depth error [cm] ↓		Median depth error [cm] ↓		Mean disparity error [pix] ↓		1PA [%] ↑	
		Split 1	Split 3	Split 1	Split 3	Split 1	Split 3	Split 1	Split 3
EIS [37]	✗	<u>13.7</u>	22.4	-	-	-	-	89.0	88.1
EITNet [2]	✓	14.2	19.4	<u>5.9</u>	10.4	0.55	0.75	<u>92.1</u>	<u>89.6</u>
DDES [51]	✓	16.7	27.8	6.8	14.7	0.59	0.94	89.8	74.8
DTC-PDS	✓	15.3	<u>18.6</u>	6.4	<u>8.7</u>	0.56	<u>0.65</u>	91.5	88.7
CTC-PDS	✓	14.9	20.6	6.4	10.6	<u>0.53</u>	0.73	91.6	88.2
DTC-SPADE	✓	**13.5**	**17.1**	**5.2**	**7.9**	**0.46**	**0.60**	**93.0**	**89.7**

Table 2. Results for dense disparity estimation. The blank entries denote the unavailability of the respective values from the associated paper. In each criterion, we denote the best result in bold and the second best with an underline. DTC-SPADE outperforms all the other methods on all metrics. EO denotes event-only input for both training and inference. Note that EITNet requires gray-scale images for training but not for inference.

trast, the event queue method mainly utilizes recent information, with much larger weights on recent events than distant ones. Note that SBT merges for each frame 10 ms of events, which is highly sparse. However, DTC learns to aggregate past information through training and forms dense feature maps, as depicted in Fig. 2.

3.2. Empirical results

In this subsection, we compare DTC-PDS and DTC-SPADE with other state-of-the-art event-based stereo matching methods. The results are presented in table 2. Values of other models are taken from their papers. As event-only approaches, both of our models outperform DDES [51]

in all criteria. DTC-SPADE even surpasses methods trained with additional intensity images. Note that DTC-PDS also outperforms EITNet on split 3 except for the 1PA criterion. The results demonstrate that SPADE can improve the network performance, which is supported by further multiple random seeds experiments (see supplement material). The training of EIS [37] utilized both intensity images and event frames prepared with SBN, which stacks a fixed number of events to form a frame. When the temporal density of events fluctuates it can cause potential time misalignment between events and ground truth disparities. This could be the reason of its poor performance. The EITNet [2] trained an image-reconstruction sub-network and used structural information from the image to strengthen the quality of estimated disparities. This approach is a double sword since the network's performance will be harmed if the reconstruction quality is sub-optimal. It also tremendously increases the network's training time as well as the computational cost at inference, thus hardly applicable for high-speed scenarios. DTC-SPADE has comparatively much lower computational cost. The network extracts underlying edge information from instant event frames, enabling it to render finer local structures compared to DDES and DTC-PDS, as Fig. 4 shows. Various intrinsic dynamics of DTC also enables the network to generalize well on temporally highly dynamical event streams.

3.3. Streaming experiment

In real-world applications, events are assumed to be generated consecutively by the sensor with variable duration. It would be redundant for algorithms to repetitively review a fixed length of past information to produce accurate disparities, which is by far the standard training setup for our model and all the other methods we have compared with. To test the robustness and real-time applicability of DTC-PDS and DTC-SPADE, we designed a set of streaming experiments. In these experiments, the entire test split is continuously fed into the model, which evolves an equal length of steps and estimates the corresponding disparities. The results are summarized in table 3. DTC-PDS (DTC-SPADE) reaches an inference speed of 110 (64) FPS (in a single NVIDIA Tesla V100 32G GPU) with a similar level of accuracy as when testing based on a fixed length of preceding event frames. To our best knowledge, our models are the first to perform streaming experiments for dense disparity estimation on the MVSEC dataset. More details about the experiments, including the FPS calculation are in the supplement material.

3.4. DSEC dataset

To demonstrate the general usage and robustness of our method. We further trained our models on the recently proposed DSEC dataset [16]. Preliminary experiments (table 4) showed that DTC-SPADE achieved state-of-the-art performance among event-only methods. More details of the experiments are in the supplement material.

Method	DTC-PDS		DTC-SPADE	
	Split 1	Split 3	Split 1	Split 3
1PA ↑	91.2(91.5)	88.7(88.7)	92.9(93.0)	89.6(89.8)
MDE ↓	15.1(15.3)	18.6(18.6)	13.5(13.5)	17.1(17.1)
FPS	110		64	

Table 3. Results for streaming experiment. Values in standard testing setup are in brackets.

Method	EO	1PE ↓	2PE ↓	MAE ↓	RMSE ↓
EIS-EI [37]	✗	5.814	1.055	0.396	0.905
EIS-ES [37]	✓	9.958	2.645	0.529	1.222
DDES [51]	✓	10.915	2.905	0.576	1.386
DTC-PDS	✓	9.517	2.356	0.527	1.264
DTC-SPADE	✓	9.27	2.405	0.526	1.285

Table 4. Comparison on the DSEC dataset. Results are also released on the DSEC disparity benchmark website [1].

4. Conclusion

In this study, we proposed a novel encoding method for sparse events, CTC, which encodes event streams with trainable intrinsic dynamics and low computational cost, and its abstracted version DTC to accelerate simulation. We demonstrated their advantages over other methods on the event-based stereo matching task on the MVSEC dataset. On the network level, we proposed an efficient dual-path architecture, DTC-SPADE, where underlying edge information from event frames is utilized to enhance estimation quality. We demonstrated the superiority of our model over existing state-of-the-art works on both the MVSEC and the DSEC dataset. In streaming experiments, our models exhibit robustness and fast inference, showing a great potential for real-time applications in high-speed scenario. Inspired from biological neurons, both CTC and DTC show clear advantages over other methods in terms of accuracy, model size and speed when running on GPU.An extension of our principles to SNNs implemented in neuromorphic hardware [10, 11, 14, 24, 33, 43] could further lead to super fast event-based stereo system [3]. Our encoding method can be applied to other event-based tasks as well, such as optical flow estimation and slam. Our fast and lightweight stereo network can also benefit downstream applications such as 3D object detection.

Acknowledgement

This work is partially supported by the National Key Research and Development Program of China (2021YFF1200800)

References

[1] Dsec disparity benchmark. https://dsec.ifi.uzh.ch/uzh/disparity-benchmark/. 8

[2] Soikat Hasan Ahmed, Hae Woong Jang, SM Nadim Uddin, and Yong Ju Jung. Deep event stereo leveraged by event-to-image translation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 882–890, 2021. 2, 3, 4, 5, 6, 7, 8

[3] Alexander Andreopoulos, Hirak J Kashyap, Tapan K Nayak, Arnon Amir, and Myron D Flickner. A low power, high throughput, fully event-based stereo system. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7532–7542, 2018. 8

[4] Nicolas Ballas, Li Yao, Chris Pal, and Aaron Courville. Delving deeper into convolutional networks for learning video representations. *arXiv preprint arXiv:1511.06432*, 2015. 2

[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 communications*, 11(1):1–15, 2020. 2

[6] Pablo Rodrigo Gantier Cadena, Yeqiang Qian, Chunxiang Wang, and Ming Yang. Spade-e2vid: Spatially-adaptive denormalization for event-based video reconstruction. *IEEE Transactions on Image Processing*, 30:2488–2500, 2021. 5

[7] Marco Cannici, Marco Ciccone, Andrea Romanoni, and Matteo Matteucci. Asynchronous convolutional networks for object detection in neuromorphic cameras. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 0–0, 2019. 2

[8] Yongqiang Cao, Yang Chen, and Deepak Khosla. Spiking deep convolutional neural networks for energy-efficient object recognition. *International Journal of Computer Vision*, 113(1):54–66, 2015. 2

[9] Haosheng Chen, David Suter, Qiangqiang Wu, and Hanzi Wang. End-to-end learning of object motion estimation from retinal events for event-based object tracking. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 10534–10541, 2020. 2

[10] Benjamin Cramer, Sebastian Billaudelle, Simeon Kanya, Aron Leibfried, Andreas Grübl, Vitali Karasenko, Christian Pehle, Korbinian Schreiber, Yannik Stradmann, Johannes Weis, et al. Surrogate gradients for analog neuromorphic computing. *Proceedings of the National Academy of Sciences*, 119(4), 2022. 8

[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. *Proceedings of the IEEE*, 109(5):911–934, 2021. 8

[12] Wei Fang, Zhaofei Yu, Yanqi Chen, Timothée Masquelier, Tiejun Huang, and Yonghong Tian. Incorporating learnable membrane time constant to enhance learning of spiking neural networks. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2661–2671, 2021. 2

[13] Ken-ichi Funahashi and Yuichi Nakamura. Approximation of dynamical systems by continuous time recurrent neural networks. *Neural networks*, 6(6):801–806, 1993. 2, 3

[14] Steve B Furber, Francesco Galluppi, Steve Temple, and Luis A Plana. The spinnaker project. *Proceedings of the IEEE*, 102(5):652–665, 2014. 8

[15] Daniel Gehrig, Antonio Loquercio, Konstantinos G Derpanis, and Davide Scaramuzza. End-to-end learning of representations for asynchronous event-based data. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5633–5643, 2019. 2

[16] Mathias Gehrig, Willem Aarents, Daniel Gehrig, and Davide Scaramuzza. Dsec: A stereo event camera dataset for driving scenarios. *IEEE Robotics and Automation Letters*, 6(3):4947–4954, 2021. 3, 5, 8

[17] Ramin Hasani, Mathias Lechner, Alexander Amini, Daniela Rus, and Radu Grosu. A natural lottery ticket winner: Reinforcement learning with ordinary neural circuits. In *International Conference on Machine Learning*, pages 4082–4093. PMLR, 2020. 4

[18] Ramin Hasani, Mathias Lechner, Alexander Amini, Daniela Rus, and Radu Grosu. Liquid time-constant networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 7657–7666, 2021. 2, 3, 4

[19] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997. 2

[20] Alex Kendall, Hayk Martirosyan, Saumitro Dasgupta, Peter Henry, Ryan Kennedy, Abraham Bachrach, and Adam Bry. End-to-end learning of geometry and context for deep stereo regression. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 66–75, 2017. 1

[21] Seijoon Kim, Seongsik Park, Byunggook Na, and Sungroh Yoon. Spiking-yolo: Spiking neural network for energy-efficient object detection. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 11270–11277, 2020. 2

[22] Agnes Korcsak-Gorzo, Michael G Müller, Andreas Baumbach, Luziwei Leng, Oliver J Breitwieser, Sacha J van Albada, Walter Senn, Karlheinz Meier, Robert Legenstein, and Mihai A Petrovici. Cortical oscillations support sampling-based computations in spiking neural networks. *PLoS computational biology*, 18(3):e1009753, 2022. 2

[23] Alexander Kugele, Thomas Pfeil, Michael Pfeiffer, and Elisabetta Chicca. Efficient processing of spatio-temporal data streams with spiking neural networks. *Frontiers in Neuroscience*, 14:439, 2020. 2

[24] Akos F Kungl, Sebastian Schmitt, Johann Klähn, Paul Müller, Andreas Baumbach, Dominik Dold, Alexander Kugele, Eric Müller, Christoph Koke, Mitja Kleider, et al. Accelerated physical emulation of bayesian inference in spiking neural networks. *Frontiers in neuroscience*, page 1201, 2019. 8

[25] Xavier Lagorce, Garrick Orchard, Francesco Galluppi, Bertram E Shi, and Ryad B Benosman. Hots: a hierarchy of event-based time-surfaces for pattern recognition. *IEEE transactions on pattern analysis and machine intelligence*, 39(7):1346–1359, 2016. 1, 2

[26] Mathias Lechner, Ramin Hasani, Alexander Amini, Thomas A Henzinger, Daniela Rus, and Radu Grosu. Neural circuit policies enabling auditable autonomy. *Nature Machine Intelligence*, 2(10):642–652, 2020. 2

[27] Chankyu Lee, Adarsh Kumar Kosta, Alex Zihao Zhu, Kenneth Chaney, Kostas Daniilidis, and Kaushik Roy. Spike-flownet: event-based optical flow estimation with energy-efficient hybrid neural networks. In *European Conference on Computer Vision*, pages 366–382. Springer, 2020. 2

[28] Luziwei Leng, Roman Martel, Oliver Breitwieser, Ilja Bytschok, Walter Senn, Johannes Schemmel, Karlheinz Meier, and Mihai A Petrovici. Spiking neurons with short-term synaptic plasticity form superior generative networks. *Scientific reports*, 8(1):1–11, 2018. 2

[29] Ming Liang and Xiaolin Hu. Recurrent convolutional neural network for object recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3367–3375, 2015. 2

[30] Wolfgang Maass. Networks of spiking neurons: the third generation of neural network models. *Neural networks*, 10(9):1659–1671, 1997. 2

[31] Ana I Maqueda, Antonio Loquercio, Guillermo Gallego, Narciso García, and Davide Scaramuzza. Event-based vision meets deep learning on steering prediction for self-driving cars. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5419–5427, 2018. 1, 2

[32] Nikolaus Mayer, Eddy Ilg, Philip Hausser, Philipp Fischer, Daniel Cremers, Alexey Dosovitskiy, and Thomas Brox. A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4040–4048, 2016. 1

[33] 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. 8

[34] Nico Messikommer, Daniel Gehrig, Antonio Loquercio, and Davide Scaramuzza. Event-based asynchronous sparse convolutional networks. In *European Conference on Computer Vision*, pages 415–431. Springer, 2020. 2

[35] Diederik Paul Moeys, Federico Corradi, Emmett Kerr, Philip Vance, Gautham Das, Daniel Neil, Dermot Kerr, and Tobi Delbrück. Steering a predator robot using a mixed frame/event-driven convolutional neural network. In *2016 Second International Conference on Event-based Control, Communication, and Signal Processing (EBCCSP)*, pages 1–8. IEEE, 2016. 1

[36] Mohammad Mostafavi, Lin Wang, and Kuk-Jin Yoon. Learning to reconstruct hdr images from events, with applications to depth and flow prediction. *International Journal of Computer Vision*, 129(4):900–920, 2021. 1, 3

[37] Mohammad Mostafavi, Kuk-Jin Yoon, and Jonghyun Choi. Event-intensity stereo: Estimating depth by the best of both worlds. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4258–4267, 2021. 2, 4, 6, 7, 8

[38] 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 Processing Magazine*, 36(6):51–63, 2019. 2

[39] Daniel Neil, Michael Pfeiffer, and Shih-Chii Liu. Phased lstm: Accelerating recurrent network training for long or event-based sequences. *arXiv preprint arXiv:1610.09513*, 2016. 2

[40] Anh Nguyen, Thanh-Toan Do, Darwin G Caldwell, and Nikos G Tsagarakis. Real-time 6dof pose relocalization for event cameras with stacked spatial lstm networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 0–0, 2019. 1

[41] Garrick Orchard, Cedric Meyer, Ralph Etienne-Cummings, Christoph Posch, Nitish Thakor, and Ryad Benosman. Hfirst: A temporal approach to object recognition. *IEEE transactions on pattern analysis and machine intelligence*, 37(10):2028–2040, 2015. 2

[42] Taesung Park, Ming-Yu Liu, Ting-Chun Wang, and Jun-Yan Zhu. Semantic image synthesis with spatially-adaptive normalization. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 2337–2346, 2019. 2, 5

[43] J. Pei, L. Deng, S. Song, M. Zhao, and L. Shi. Towards artificial general intelligence with hybrid tianjic chip architecture. *Nature*, 572(7767):106, 2019. 8

[44] Mihai Alexandru Petrovici. *Form versus function: theory and models for neuronal substrates*. Springer, 2016. 3

[45] Mihai A Petrovici, Johannes Bill, Ilja Bytschok, Johannes Schemmel, and Karlheinz Meier. Stochastic inference with spiking neurons in the high-conductance state. *Physical Review E*, 94(4):042312, 2016. 2

[46] WH Pressa, SA Teukolsky, WT Vetterling, and BP Flannery. Numerical recipes 3rd edition: The art of scientific computing, 2007. 3

[47] Cedric Scheerlinck, Nick Barnes, and Robert Mahony. Asynchronous spatial image convolutions for event cameras. *IEEE Robotics and Automation Letters*, 4(2):816–822, 2019. 2

[48] René Schuster, Oliver Wasenmuller, Christian Unger, and Didier Stricker. Sdc-stacked dilated convolution: A unified descriptor network for dense matching tasks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2556–2565, 2019. 2, 5

[49] Alex Sherstinsky. Deriving the recurrent neural network definition and rnn unrolling using signal processing. In *Critiquing and Correcting Trends in Machine Learning Workshop at Neural Information Processing Systems*, volume 31, 2018. 4

[50] Sumit Bam Shrestha and Garrick Orchard. Slayer: Spike layer error reassignment in time. *arXiv preprint arXiv:1810.08646*, 2018. 2

[51] Stepan Tulyakov, Francois Fleuret, Martin Kiefel, Peter Gehler, and Michael Hirsch. Learning an event sequence embedding for dense event-based deep stereo. In *Proceedings*

of the IEEE/CVF International Conference on Computer Vision, pages 1527–1537, 2019. 2, 3, 4, 5, 6, 7, 8

[52] Stepan Tulyakov, Anton Ivanov, and Francois Fleuret. Practical deep stereo (pds): Toward applications-friendly deep stereo matching. *arXiv preprint arXiv:1806.01677*, 2018. 1, 2, 5

[53] Lin Wang, Yo-Sung Ho, Kuk-Jin Yoon, et al. Event-based high dynamic range image and very high frame rate video generation using conditional generative adversarial networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10081–10090, 2019. 1

[54] Yujie Wu, Lei Deng, Guoqi Li, Jun Zhu, Yuan Xie, and Luping Shi. Direct training for spiking neural networks: Faster, larger, better. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 1311–1318, 2019. 2

[55] 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 processing systems*, pages 802–810, 2015. 2

[56] Haofei Xu and Juyong Zhang. Aanet: Adaptive aggregation network for efficient stereo matching. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1959–1968, 2020. 1

[57] Zheyu Yang, Yujie Wu, Guanrui Wang, Yukuan Yang, Guoqi Li, Lei Deng, Jun Zhu, and Luping Shi. Dashnet: A hybrid artificial and spiking neural network for high-speed object tracking. *arXiv preprint arXiv:1909.12942*, 2019. 2

[58] C Ye, A Mitrokhin, C Parameshwara, C Fermüller, JA Yorke, and Y Aloimonos. Unsupervised learning of dense optical flow and depth from sparse event data. corr abs/1809.08625 (2018), 1809. 1

[59] Feihu Zhang, Victor Prisacariu, Ruigang Yang, and Philip HS Torr. Ga-net: Guided aggregation net for end-to-end stereo matching. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 185–194, 2019. 1

[60] Wenrui Zhang and Peng Li. Temporal spike sequence learning via backpropagation for deep spiking neural networks. *arXiv preprint arXiv:2002.10085*, 2020. 2

[61] Yi Zhou, Guillermo Gallego, Henri Rebecq, Laurent Kneip, Hongdong Li, and Davide Scaramuzza. Semi-dense 3d reconstruction with a stereo event camera. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 235–251, 2018. 2

[62] Alex Zihao Zhu, Yibo Chen, and Kostas Daniilidis. Real-time time synchronized event-based stereo. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 433–447, 2018. 2, 3, 5

[63] Alex Zihao 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 Robotics and Automation Letters*, 3(3):2032–2039, 2018. 2, 5

[64] Alex Zihao Zhu, Liangzhe Yuan, Kenneth Chaney, and Kostas Daniilidis. Ev-flownet: Self-supervised optical flow estimation for event-based cameras. *arXiv preprint arXiv:1802.06898*, 2018. 1, 2, 5

[65] Alex Zihao Zhu, Liangzhe Yuan, Kenneth Chaney, and Kostas Daniilidis. Unsupervised event-based learning of optical flow, depth, and egomotion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 989–997, 2019. 2