This CVPR paper is the Open Access version, provided by the Computer Vision Foundation.
Except for this watermark, it is identical to the accepted version;
the final published version of the proceedings is available on IEEE Xplore.

Graph Neural Network Combining Event Stream and Periodic Aggregation for Low-Latency Event-based Vision

Manon Dampfhoffer[1], Thomas Mesquida[1], Damien Joubert[2], Thomas Dalgaty[1],
Pascal Vivet[1], Christoph Posch[2]

[1]Univ. Grenoble Alpes, CEA, List, F-38000 Grenoble, France. [2]Prophesee, Paris, France.

{manon.dampfhoffer, thomas.mesquida}@cea.fr cposch@prophesee.ai

Abstract

Event-based cameras asynchronously detect changes in light intensity with high temporal resolution, making them a promising alternative to RGB camera for low-latency and low-power optical flow estimation. However, state-of-the-art convolutional neural network methods create frames from the event stream, therefore losing the opportunity to exploit events for both sparse computations and low-latency prediction. On the other hand, asynchronous event graph methods could leverage both, but at the cost of avoiding any form of time accumulation, which limits the prediction accuracy. In this paper, we propose to break this accuracy-latency trade-off with a novel architecture combining an asynchronous accumulation-free event branch and a periodic aggregation branch. The periodic branch performs feature aggregations on the event graphs of past data to extract global context information, which improves accuracy without introducing any latency. The solution could predict optical flow per event with a latency of tens of microseconds on asynchronous hardware, which represents a gain of three orders of magnitude with respect to state-of-the-art frame-based methods, with 48x less operations per second. We show that the solution can detect rapid motion changes faster than a periodic output. This work proposes, for the first time, an effective solution for ultra low-latency and low-power optical flow prediction from event cameras.

1. Introduction

Optical flow is the apparent relative motion of objects and surfaces in a scene (which could come from the motion of an object or from the observer itself) and is used in many downstream computer vision tasks, such as video stabilization [35], virtual reality [25], navigation [36] and obstacle avoidance [32]. Using an RGB camera, it can be measured as the pixel's displacement between two consecutive

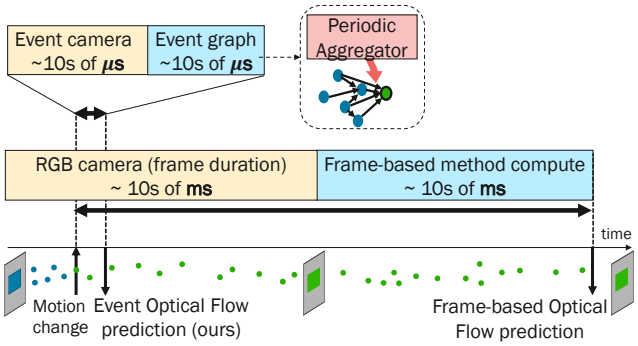


Figure 1. Ultra-low latency optical flow prediction with asynchronous event graph neural networks and periodic aggregator.

frames. However, the latency of the optical flow prediction is limited, both by the frame rate of the sensor and the computational time of the prediction algorithm. Therefore, rapid motion changes cannot be detected during this blind time, which can be critical in some real-time applications such as navigation [12]. The latency of the sensor can be decreased by increasing the frame rate, which comes at a high computational cost and can cause bandwidth problems.

Event cameras are a promising alternative to RGB cameras in this context [11]. The pixels of event cameras function asynchronously and produce an event whenever a change in the light intensity is detected, at a temporal resolution of microseconds [17]. Therefore, they are particularly adapted for detecting fast motion changes with low latency. However, state-of-the-art convolutional neural networks (CNN) cannot handle the asynchronous flow of events and construct frames from events [7, 9, 13, 20, 21, 24, 30, 33, 38], loosing the opportunity to exploit the event information at low latency. Moreover, state-of-the-art methods are computationally-expensive [13, 20], limiting real-time implementation on embedded hardware with constrained resources.

Therefore, the concept of event graphs (based on Graph

Neural Networks - GNN) has emerged as a solution [5]. Events are represented in a continuously evolving graph, asynchronously triggering graph convolutions at the arrival of new events, that can allow event-level predictions at ultra-low latency [6, 12, 26]. This has motivated the creation of new asynchronous hardware to support these implementations [14, 34]. Moreover, by constraining edges of the graphs to be directed, from past-to-future only (preventing backward connections), the HUGNet approach [6] removes the additional latency of graph update for each new events, allowing ultra-low latency graph building. However, the latency constraint also prevents from using any form of accumulations in the GNN (e.g. pooling layers or normalizations on future data). This limits the aggregation of global context information, which is important for understanding the scene dynamics, and hence severely impacts the accuracy of the prediction.

In this paper, we propose to break this accuracy-latency trade-off and leverage both asynchronous event graphs and periodic temporal accumulations. We propose a novel neural network architecture combining an accumulation-free asynchronous event branch and a periodic aggregation branch performing feature aggregations on the event graphs of past data. Therefore, the periodic branch can extract global context information, improving the accuracy of optical flow estimation, without increasing the prediction latency compared to the event branch alone. However, the periodic branch, only having access to past data, cannot accurately detect fast motion changes, and hence merging the periodic outputs with the event branch outputs is crucial. The proposed method could predict optical flow from event cameras with a total latency in the order of 10s of μs in asynchronous hardware such as [34], while a standard frame-based approach would have a latency in the order of 10s of ms (Fig. 1). We further study if this reduction in the prediction latency also translates into a more accurate prediction of fast-changing optical flow. We show that, depending on the speed of change and the aggregation period of the periodic branch, the event prediction can outperform the periodic prediction at low latency by up to 59%.

Our contributions can be summarized as follows:

- We propose a new version of the HUGNet [6] event graph, *HUGNet2*, entirely accumulation-free, and hence compatible with ultra low-latency per event prediction.
- We introduce a new architecture combining HUGNet2 with a Periodic Aggregator (PA), *HUGNet2+PA*, which largely improves the overall prediction accuracy compared to HUGNet2 alone, while introducing no latency.
- Our architecture can effectively leverage the most recent events in the event branch to detect a rapid motion change faster than a periodic output, and outperforms the state-of-the-art on the challenging Rock Scenes dataset [6].
- The latency of our method is reduced by 3 orders of magnitude compared to the state-of-the-art (from 10s of ms to 10s of μs on asynchronous hardware), while also reducing number of operations (up to 48x) and parameters.

2. Related Works

Event-based Optical Flow Estimation. Moving edges are locally described in event data streams as 3D structures that can be analytically estimated online [3] or optimized offline using contrast maximization techniques [28, 29]. Assuming a constant speed, events can be grouped into features tracked over time that enable local motion estimations [18, 27]. Motion descriptors can also be extracted out of events using bio-inspired spiking handcrafted patterns [22]. However, all these techniques suffer from poor performance when the event stream is complex due to noise or textured objects. Inspired by image-based optical flow estimation, neural network methods create event frames, by counting events in each pixel during a certain temporal window, that can be processed by CNNs [7, 9, 13, 20, 21, 24, 30, 33, 38]. In addition, EV-FlowNet [38] use the last event timestamp in another input channel. STE-Flow [9] propose to adapt the event frame rate to the event density (i.e. each frame contains the same number of events), which allows a more accurate representation of high speed motions. ADM-Flow [20] introduce a learnable adaptive density module which can compensate for heterogeneous event density between event frames. Furthermore, Spiking Neural Networks [16, 31] have been proposed, as spiking computations can also be realized in an event-based manner on asynchronous hardware.

Event Graph Neural Networks. Frame-based representations discard the precise temporal resolution of the event and can neither fully exploit the sparsity of events for sparse computations, nor the low latency of event cameras for low latency prediction. In order to overcome these issues, recent methods have proposed to represent event data as a graph, each event being represented as a node with connectivity according to spatio-temporal proximity, that can be processed by a GNN, for object classification [4, 8, 26, 34], object detection [12, 26], motion segmentation [23], or optical flow estimation [6]. Notably, event graphs can also predict at the node-level (i.e. event-level), which can be exploited in dense tasks such as optical flow estimation. Indeed, a prediction per event could allow to estimate locally the optical flow at low latency. However, fully leveraging this low-latency opportunity requires to impose constraints on the event graph operations. Indeed, when an event arrives, all the elements required for computing its output must be already available (without waiting), such that only the time of performing the graph convolution is required, which can be realized in microseconds [34]. In this context, [6] proposes the Hemi-spherical Update Graph method (HUGNet). At each new event,

edges are constructed only on a hemi-spherical neighborhood of past events (i.e. the graph is directed, with past-to-future connections only), which reduces the latency of graph update to zero. However, current event graph methods [4, 6, 8, 23, 26, 34], use operations in the GNN relying on future data (e.g. pooling layers, graph-level normalization with on-the-fly statistics, or normal vectors across past and future events), that limits the latency of event-level prediction. Notably, [12] converts all GNN operations to an event-based version, showing for the first time a low-latency solution for an object detection application. In particular, they propose an event graph pooling that can be made asynchronous by keeping only directed edges of pooled nodes, but at the cost of severe accuracy degradation compared to a standard event graph pooling.

3. Methods

3.1. Event Graph and Periodic Aggregator

We describe the proposed architecture composed of an asynchronous event-based branch and a periodic branch (Fig. 2). The event branch contains the graph building, a GNN (used as a feature extractor for both event and periodic branches), and an Event head. In the event branch, the computations are asynchronously triggered as events arrive, which allow sparse computations (proportional to the number of events), and ultra low-latency per event prediction. The periodic branch operates in parallel of the event branch and is composed of a PA module and a Periodic head that periodically outputs dense optical flow prediction. Note that the method only use event data (no images).

3.1.1. Event Branch

Events are output asynchronously from event cameras as (x, y, t, p) features, corresponding to x and y pixel coordinates, the timestamp t of the emission of the event, and its polarity p (positive or negative, if the event corresponded to an increase or decrease of light intensity). A graph is represented as $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, with $\mathcal{V}$ being a set of vertices, and $\mathcal{E}$ a set of edges. The vertices (and edges) are associated with features, on which GNN layers are applied to compute output features. In an event graph, events are represented as vertices, and edges are formed between events based on spatio-temporal proximity, commonly using a k Nearest Neighbors search (a node is connected to its k closest nodes in the spatio-temporal 3D space) [6, 12, 34]. For enabling ultra-low latency prediction, we adopt the HUGNet [6] graph building, which constrains edges to be past-to-future directed, i.e. on recent past events only (that are already in memory). This means that the k Nearest Neighbors search is constrained to the hemi-spherical space around the event in (-r_{xy},+r_{xy}) for (x, y) coordinates and (-r_t,0) for t, r_{xy} being the pixel radius and r_t the temporal radius.

For the GNN, we propose an improved version of HUGNet [6], called *HUGNet2*, that is compatible with low latency prediction. Indeed, as event graphs are highly heterogeneous, using instance normalization with on-the-fly statistics greatly improves the accuracy. However, this requires processing the graph over the entire sequence before any prediction can be made. Moreover, HUGNet uses normal vectors computed on a 3D neighborhood (including future events) around the event, which would also add latency. Therefore, HUGNet2 does not use neither normalization nor normal vectors. In addition, HUGNet2 replaces the SplineConv [10] in the first GNN layer, by a PointTransformerConv [37], which slightly improves the accuracy while reducing the number of parameters.

GNN and PA outputs are merged in the Event branch. However, the PA output for the period T can only be available with a delay of 2 periods (Fig. 2c). Indeed, the computation can only start after the end of the accumulation period, and the PA module can take up to another frame to be computed (not more to allow real-time processing, but not less to reduce power consumption). That is why, in order to introduce no latency to the event branch, the merge is done with PA output immediately available when the event arrive (therefore corresponding to past events with a delay of 2 periods). In particular, the merge uses the output features of the event (computed in the GNN) and the PA(T-2) features in the pixel location of the event.

The merged features are fed to the Event head, which predicts per event optical flow. This temporal shift may degrade the accuracy of the event prediction for fast motions. However, we will show that the Event head can leverage the event-level information contained in the GNN output features to faster detect motion changes. Note that, for training on a GPU, the graphs are built in advance and GNN layers are computed on an entire graph in parallel. However, on asynchronous hardware, the graph building and Event branch will be computed on-the-fly as events arrive (following the computation flow described in Fig. 2b).

3.1.2. Periodic Branch

The event branch has no pooling layers (which would require temporal accumulation and hence would introduce latency in the event computation flow) and therefore has a limited field of view. The goal of the periodic branch is to perform broader spatio-temporal aggregation of features allowing a better understanding of the global scene. Therefore , the PA module accumulates the output features of the GNN during a certain period, on which a spatio-temporal pooling operation is performed. This means that the pooling operation is performed on the features of the events accumulated during the period for each spatial location (temporal and spatial resolutions can be adjusted). The pooled features are processed by the PA module, composed of convolutional and recurrent layers to perform the spatio-temporal

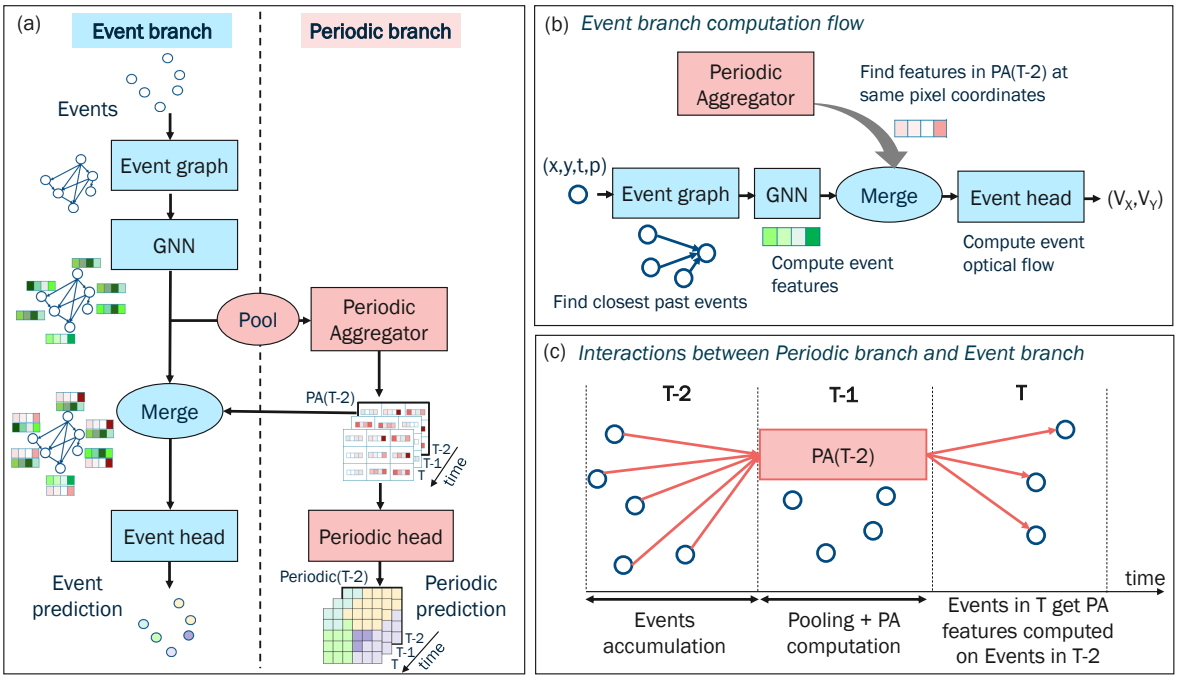


Figure 2. (a) Proposed architecture (HUGNet2+PA) combining event branch and periodic branch. (b) Computation flow of the event branch, performed asynchronously at each event. (c) Interactions between events and periodic branch. The event branch takes outputs of the periodic branch computed on past events (from period T-2 for events in period T) that are immediately available (no latency).

aggregation. The PA outputs are upsampled with bilinear interpolation in order to get back to the original pixel resolution, and then processed by the Periodic head. The Periodic head outputs Periodic predictions of dense optical flow (on every pixel location).

3.2. Event Optical Flow Datasets

MVSEC. The Multi Vehicle Stereo Event Camera dataset (MVSEC) [38, 39] includes indoor scenes from a flying drone and outdoor driving scenes, with 346x260 pixels resolution and groundtruth optical flow labels at 20 Hz. As the most common procedure [9, 13], we train the model on the $outdoor_day2$ sequence (which lasts about 610 seconds), and evaluate the model on the $outdoor_day1$ (240s) and $indoor_flying$ sequences 1, 2 and 3 (240s). However, MVSEC scenes only contain slow and smooth motions, with a groundtruth at low frequency, and hence is not challenging enough the ability to detect fast motion changes.

Rock Scenes. Rock Scenes [6] is a synthetic dataset specifically designed for evaluating very fast motion detection. Rock Scenes data feature record cover art images moving in front of a textured background, both having an uncorrelated movement with random sharp direction changes, with groundtruth optical flow at high resolution (500 Hz) and 100x100 pixel resolution. In particular, the motion changes (direction and speed) occur in one groundtruth timestep, i.e. 2 ms. The motion changes being easy to localize in the sequences, we can observe the latency of the method detect them. The sequences last between 2 and 6 seconds, with one hundred sequences for the train split (ten objects combined with ten scenes) and four sequences for the test split (two unseen objects objects combined with 2 unseen scenes).

Data pre-processing. Event graph methods construct graphs on the raw events. However, constructing and processing graphs with large number of events is slow on GPU (which are not optimized for graph processing, contrary to the asynchronous hardware implementation that this method is targeting). Moreover event streams may contain a lot of redundant information. Therefore, following HUGNet [6], the datasets are filtered with a spatio-temporal contrast filter available in state-of-the-art event-cameras [1, 2]. In addition, MVSEC $outdoor$ sequences are downsampled with a factor of 3 such that they have a similar event density as the $indoor$ sequences. The event graphs are built for full sequences and then chopped into smaller sequences of 400ms and 800ms (with overlap of 200ms), for Rock Scenes and MVSEC respectively.

3.3. Training and Evaluation Procedure

Architecture details. For the graph building, the k Nearest Neighbor search is perfomed on a hemi-sphere with radius r_{xy}= 8 and 9 pixels, and r_t= 80 and 90ms, for MVSEC and Rock Scenes respectively, with k=32 neighbors per node.

We use the pixel coordinates and polarity as node input features (not including timestamp, as the absolute time should not influence the optical flow prediction and may lead to overfitting). HUGNet2 is composed of a PointTransformerConv [37] layer followed by 4 GCNConv [15] layers, each with 32 output features. Note that PointTransformerConv is more costly than GCNConv (which simply performs mean aggregations of the projected features of the neighbors), and hence is only used in the first layer where the input size is small. The output of the 5 layers are concatenated and fed to a fully connected layers of output size 64. The periodic branch then performs a feature-wise max pooling on the 64 output features of the GNN, in a pixel grid of 1/4th the spatial resolution of the original one in each dimension (the temporal resolution corresponds to the PA rate and is a parameter of study). The PA module is composed of a 2D convolutional layer (stride 2, 3x3 kernel, 64 channels), a SepConvGRU layer as in E-RAFT [13] (2D recurrent convolutions with 1x5 kernel performed first horizontally then vertically) with 64 output channels, and a 2D convolutional layer (3x3 kernel, 128 channels). The merge in the event branch simply concatenates the 64 GNN output features of a node with the 128 PA(T-2) output features at the same pixel coordinates. The Event and Periodic head are composed of fully connected layers (192-128-64-2 and 128-2, respectively), on each event (each pixel, respectively).

Training details. AdamW [19] optimizer is used with batch size of 1, with learning rate starting at 5e-4 and divided by 2 if no relative change of 0.05 in the loss is observed during 10 epochs. The model is trained in a supervised manner using a smooth-L1-loss (as in HUGNet [6]) for both Event predictions and Periodic predictions. Data augmentation is used: random horizontal and vertical flips, temporal stretching with a factor between 0.5 and 1.5, random rotations from 0 to 360 degrees, and edge dropout (removing edges of the graph) with probability 0.25. In addition for MVSEC, random cropping of size 256x256 is used. As instance normalization with on-the-fly statistics could not be computed at low latency, it is only used as a warm-up for 100 epochs, and then removed while the models are fine-tuned for 100 more epochs. Experiments are performed on a NVIDIA A100 GPU with 40GB memory.

Evaluation metrics. The Endpoint Error is defined as the L2-norm of the difference of the predicted vs. groundtruth optical flow [6, 13, 20, 38]. We evaluate the Sparse Endpoint Error (i.e. on pixels where there is at least an event) for both Event prediction (selecting the last event per groundtruth voxel) and Periodic prediction (upsampling or downsampling the prediction to match the groundtruth rate). In frame-based methods, the number of operations per second is fixed (number of operations per frames x frame rate). In event graph methods, the number of operations per event is fixed, but the number of events varies with time, therefore we use the average event density. The proposed architecture has both operations per event and fixed operations per period, in the event and periodic branches, respectively. The latency of our method on dedicated asynchronous hardware is estimated based on the FPGA implementation for event graph building and processing from [34], reporting a latency of 16 μs per event. We estimate our latency at $\approx$ 50 μs, considering that our graph building is similar and our number of operations per event in the GNN is 3x higher than theirs (and so the latency will be increased proportionally, given similar hardware resources and configuration). We estimate the latency of HUGNet [6] superior to 400 ms, as the method uses normalization with graph-level statistics, and hence the prediction cannot be made before the end of the sequence (400 ms in their experiments). The latency of frame-based methods is estimated to be in between 1 to 2 groundtruth frames, depending on the speed of the computation, similar to our periodic prediction (as explained previously in Fig. 2c).

4. Results

4.1. Comparison with State-of-the-Arts

We benchmark the proposed approach in terms of accuracy and hardware efficiency metrics (number of parameters, operations and latency) against state-of-the-art methods on MVSEC (Table 1) and Rock Scenes (Table 2) datasets. We evaluate our proposed HUGNet2 alone, and the proposed combined architecture HUGNet2+PA with different period rates. Note that we provide the Endpoint Error of the low-latency Event prediction (and not of the Periodic prediction). On both datasets, HUGNet2 has a lower accuracy than HUGNet, but is also smaller (both in parameters and operations), and its latency is decreased by 4 orders of magnitude. Moreover, the proposed HUGNet2+PA outperforms both HUGNet2 and HUGNet with a moderate increase in the number of operations. We observe that the higher the PA frequency, the higher the accuracy, up to a certain point where the accuracy is no longer improved. This point depends on the dataset and the speed of change of optical flow with time. The rate corresponding to the groundtruth rate for MVSEC (PA_{20}), and to 1/5th of the groundtruth rate for Rock Scenes (PA_{100}) leads to the highest accuracy. On MVSEC, the proposed approach yields lower accuracy than state-of-the-art frame-based methods, however with a significant reduction in computational complexity and parameters memory (48x less operations per second and 17x less parameters than the state-of-the-art ADMFlow [20]). More importantly, we are the first to propose a solution for ultra-low latency, with 3 orders of magnitude reduction (from 10s of ms to 10s of μs) compared to previous methods. There is no state-of-the-art CNN methods on Rock Scenes, therefore we train ADMFlow [20] (at 20 Hz), the state-of-the-art

Paper	Type	Endpoint Error	Parameters	OPs/event	OPs/frame	OPs/s	Latency[b]
EV-FlowNet [38]	CNN	0.96[c]	14.1 M	-	5.4 G	107.0 G	50-100 ms
Stoffregen et al. [30]	CNN	0.64[d]	n.a.	-	n.a.	n.a.	50-100 ms
FireFlowNet [24]	CNN	1.22	0.06 M	-	5.1 G	102.8 G	50-100 ms
E-RAFT [13]	CNN	0.62	5.3 M	-	47.4 G	948.8 G	50-100 ms
STE-FlowNet [9]	CNN	0.56	n.a.	-	n.a.	n.a.	50-100 ms
DCEIFlow [33]	CNN	0.51[d]	n.a.	-	n.a.	n.a.	50-100 ms
ADMFlow [9]	CNN	0.49[d]	7.0 M	-	47.5 G	950.2 G	50-100 ms
Spike-FlowNet [16]	SNN	0.79[c]	14.1 M	-	4.4 G	88.2 G	50-100 ms
SdformerFlow [31]	SNN	0.85[d]	54.9 M	-	9.1 G	181.9 G	50-100 ms
HUGNet [6]	GNN	1.72[a]	0.14 M	0.22 M	-	17.1 G	$>$ 400 ms
HUGNet2	GNN	1.89	0.07 M	0.18 M	-	13.7 G	$\approx$ 50 μs
HUGNet2+PA$_5$	GNN + CNN	1.71	0.41 M	0.16 M	0.38 G	14.1 G	$\approx$ 50 μs
HUGNet2+PA$_{10}$	GNN + CNN	1.57	0.41 M	0.16 M	0.38 G	16.0 G	$\approx$ 50 μs
HUGNet2+PA$_{20}$	GNN + CNN	1.52	0.41 M	0.16 M	0.38 G	19.8 G	$\approx$ 50 μs

Table 1. Optical flow on **MVSEC** dataset (groundtruth at 20Hz, 76k event/s). [a]Our implementation. [b]See Section 3.3 for details. [c]Trained with self-supervised loss. [d]Trained on synthetic data.

Paper	Type	Endpoint Error	Parameters	OPs/event	OPs/frame	OPs/s	Latency[b]
ADMFlow [20] (20 Hz)	CNN	0.048[a]	7.0 M	-	12.5 G	250 G	50-100 ms
HUGNet [6]	GNN	0.037[a]	0.14 M	0.22 M	-	3.6 G	$>$ 400 ms
HUGNet2	GNN	0.066	0.07 M	0.18 M	-	2.9 G	$\approx$ 50 μs
HUGNet2+PA$_{20}$	GNN + CNN	0.040	0.41 M	0.16 M	63 M	3.8 G	$\approx$ 50 μs
HUGNet2+PA$_{50}$	GNN + CNN	0.033	0.41 M	0.16 M	63 M	5.7 G	$\approx$ 50 μs
HUGNet2+PA$_{100}$	GNN + CNN	0.030	0.41 M	0.16 M	63 M	8.8 G	$\approx$ 50 μs

Table 2. Results on **Rock Scenes** dataset (groundtruth at 500Hz, 16k event/s). [a]Our implementation. [b]See Section 3.3 for details.

method on MVSEC, on Rock Scenes, using the open source code. Notably, our method outperforms ADMFlow on this dataset, while being more efficient.

4.2. Low-latency Periodic vs. Event Predictions

We analyse the ability of the method to detect fast motion changes, and its dependence on application-related and method-related factors. In particular, we study whether the use of recent event information from the GNN and past context information from the PA can be effectively merged, such that the Event head can predict a more accurate output while improving the latency of detection of fast motion changes. Therefore, we compare the Event prediction (based on the output of the GNN and PA(T-2)) for events in period T (*Event*), with the Periodic prediction for events in period T-2 (*Periodic(T-2)*), both being available without overhead in latency (in period T, Periodic(T-2) has been already computed). Event prediction is supposed to be available instantaneously here because its latency (50 μs) is very small compared to a groundtruth timestep (2 ms).

Accuracy vs. number of operations. We evaluate the two predictions in average accuracy compared with the number of operations per second, depending on PA rate (Fig 3). Note that we compare both outputs on the Sparse Endpoint Error metric, because the Event prediction is only sparse. We also provide the respective number of operations per second per branch (Event vs. Periodic) to show their dependence on the application. Indeed, event density is higher in MVSEC, inducing more operations in the event branch, but the PA rate is smaller, decreasing the number of operations in the periodic branch, while it is the opposite in Rock Scenes. We observe a consistent error decrease (up to 59%) in the Event prediction compared to Periodic(T-2) prediction. This means that the Event head can effectively merge both types of features (GNN and PA(T-2)) and leverage the recent information contains in Event from the GNN, to improve the prediction compared to Periodic(T-2) (only using PA(T-2)). Notably, the Event prediction with low PA rate (20 / 5 Hz for Rock Scenes / MVSEC respectively) matches the accuracy of the Periodic(T-2) prediction with high PA rate (100 / 20 Hz respectively), while reducing the number of operations by 56% / 29% respectively. Moreover, the

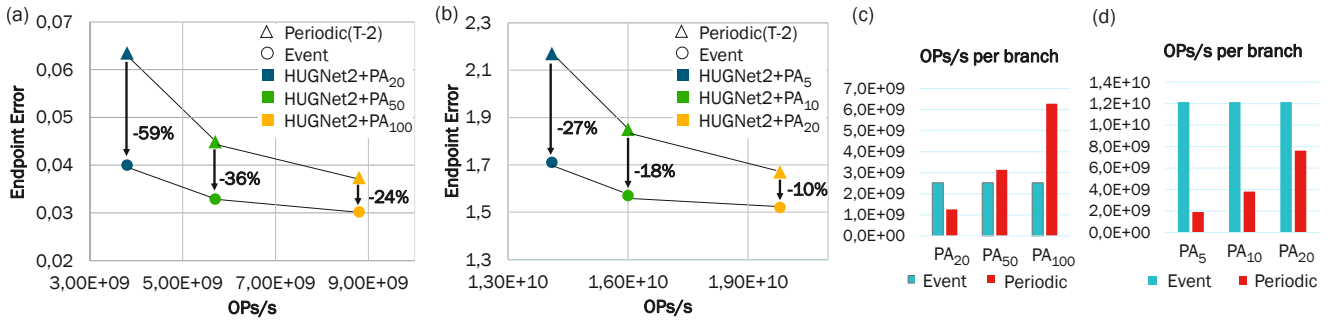


Figure 3. Endpoint Error vs. operations per seconds for the low-latency predictions (Event and Periodic(T-2)) with different PA rates for (a) Rock Scenes and (b) MVSEC. Operations per second per branch with different PA rates for (c) Rock Scenes and (d) MVSEC.

lower the PA rate, the higher the accuracy of Event compared to Periodic(T-2) prediction, as the more Periodic(T-2) is impacted by the 2 periods delay. Moreover, the gain is higher on Rock Scenes than MVSEC, because very fast motion changes occur (in one groundtruth frame, which is at 500 Hz), while changes in MVSEC are much smoother.

Latency of detection of fast motion changes. We use the Rock Scenes dataset to evaluate more specifically the ability of the Event prediction to detect when a motion change occurs. Indeed, the optical flow changes are clearly visible and abrupt (in one groundtruth timestep), therefore it is ideal to observe the reaction time of the different predictions. We show the error of Event and Periodic(T-2) predictions with time for HUGNet2+PA$_{20}$ in Fig 4, with marks indicating the precise moments of optical flow change. In addition, visualisations of the corresponding optical flow predictions and groundtruth with time are provided. We can clearly see the impact of the delay on Periodic(T-2): a motion change can be detected with a minimum of 1 to 2 frames delay (depending on whether the change occurred at the beginning or end of the frame), meaning 50 to 100 ms for the PA at 20Hz, sometimes longer. On the other hand, the Event prediction error also increases at the moment of the motion change, but then decreases faster than the Periodic prediction. Interestingly, we observe that the error of Event and Periodic branches are similar most of the time, except after a motion change. This means that the Event head can leverage GNN features on recent events to differentiate from the Periodic branch outputs when there is enough evidence that the optical flow has changed and is not yet visible in PA(T-2). However, the Event prediction takes some time to correct the optical flow prediction, which is due to accumulating enough events to be able to characterize precisely the new optical flow. In particular, we observe that, after a motion change, the Event head predicts a decrease in optical flow magnitude and then an increase in the corrected direction (which would better correspond to a real-life situation, while in the dataset, the change is instantaneous).

Comparison with ADMFlow [20]. The evolution of sparse Endpoint Error with time of the ADMFlow [20] prediction (our implementation, at 20 Hz) at T and T-2 is compared with our Event prediction in Fig. 5. Just after a change, ADMFlow(T) is better than Event, because it has access to the event data in the frame that occurred after the change. However, using ADMFlow(T-2) prediction, we observe the lag after a change (as our Periodic(T-2)). On average, our Event prediction is better than both ADMFlow(T) and ADMFlow(T-2). We hypothesize that this is partly due to to the extreme data sparsity in Rock Scenes (1.6 event/pixel/s). While ADMFlow is designed to take only 2 event aggregation periods as inputs for the prediction, our recurrent PA module functions over the entire sequence. Hence out method can accumulate information in longer periods, which improves the prediction accuracy when the motion is constant for a while. Nevertheless, the inherent limitation of frame-based methods remains valid: the lag between the emission of the event and the moment when the prediction is available impacts the latency of detection of fast motion changes.

4.3. Discussion

Our method addresses the multi-factorial trade-off between accuracy, latency and computational cost, which are impacted by many aspects. For instance, the larger the difference between the speed of motions to detect and the periodic aggregation rate, the larger the benefit of our approach leveraging events compared to a periodic output. Therefore, our method is expected to provide the largest benefits when the application requires to detect fast motions, that the frame rate of a frame-based method could not match for efficiency reasons. In this case, our method can detect those fast changes while maintaining a low rate of periodic aggregation. As shown previously, the accuracy of the Event prediction of HUGNet2+PA with PA at 5Hz (respectively 20Hz) matches the Periodic(T-2) predictions at 20Hz (respectively 100Hz) in MVSEC (respectively Rock Scenes), showing that our approach is highly efficient in terms of

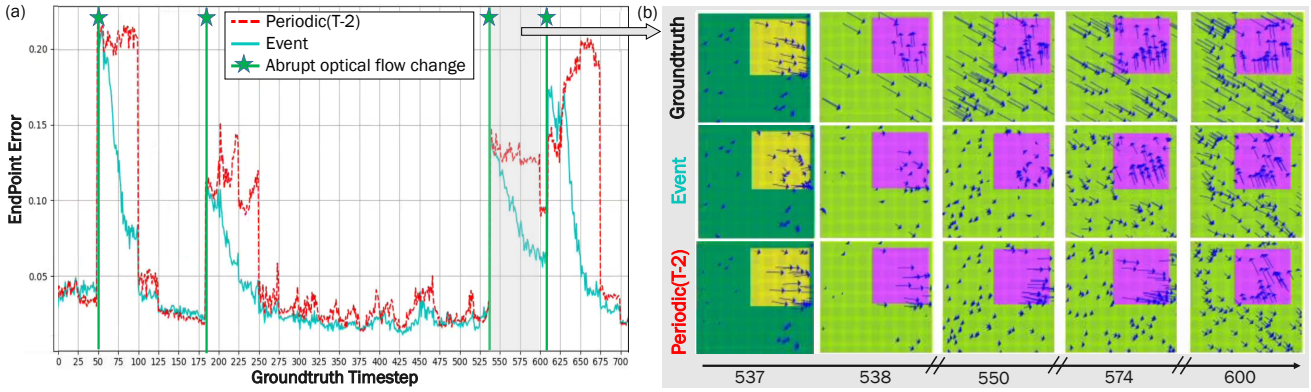


Figure 4. (a) Endpoint Error as a function of time for Event and Periodic(T-2) predictions compared to groundtruth at timestep T, with HUGNet2+PA$_{20}$, on a Rock Scenes test sequence. (b) Optical Flow predictions of Event and Periodic(T-2) vs. Groundtruth at selected groundtruth timesteps from (a). Arrows represent optical flow vectors (color masks indicate directions for the object and background).

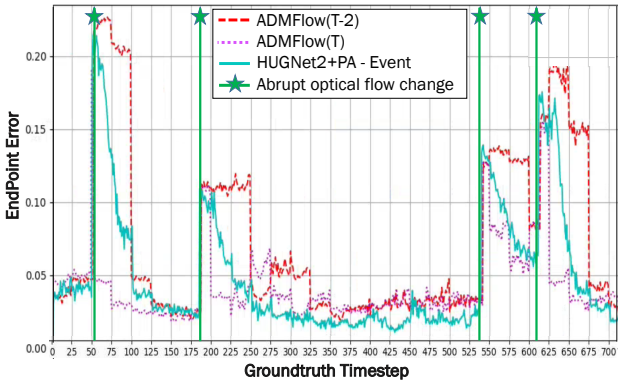


Figure 5. Endpoint error for HUGNet2+PA$_{20}$ and ADMFlow at 20 Hz on the same Rock Scenes sequence as Fig 4.

accuracy and number of operations compared to a periodic output at low latency.

Notably, our method is also impacted by the event density, which varies with time, including in similar environments, as event density depends on the motion in the scene. Indeed, higher event density can provide more information about optical flow, thus improving the accuracy, but also increases the computational cost of the Event branch. Moreover, not using event pooling in the GNN (only performing feature pooling in the Periodic branch) also limits the scalability of the GNN under power constraints, as the number of operations in every layer is impacted by the original event density. That is why our GNN uses a costly graph convolution layer only in the first layer, where the input size is small, to mitigate the number of operations. On the other hand, not pooling events also allows more predictions, as the Event branch outputs event-level predictions, which allows to faster adapt the optical flow estimation. Our PA module could be scaled for improving accuracy, by integrating more convolutional layers, other types of aggregation operations (e.g. correlation volumes), and iterative refinement [13], at the cost of more operations per frame.

Furthermore, the benefit of our method not only depends on the application constraints (such as fast changing optical flow or event density), but also on the ability of the Event head to effectively leverage both GNN and PA outputs. Indeed, the Event head should be able to detect a potential mismatch between the two types of features, indicating that recent event features should dominate over the PA features. Therefore, future work could study the choice of the merge function and the architecture of the Event head to better interpret both features. Our method could also be extended to support image inputs in addition to events, in the multimodal scenario where an event camera and RGB camera are coupled to improve the overall prediction accuracy (similar to [12]). In this case, the image inputs would be processed by the periodic branch, in addition to taking inputs from the event branch.

5. Conclusion

We propose an efficient solution for optical flow estimation from event-based cameras at ultra-low latency and with minimal computational cost. Our solution combines event graph neural networks enabling low-latency per-event prediction, with a periodic aggregator module extracting context information, improving the accuracy of the event prediction without introducing latency to the event computation flow. Remarkably, our approach is the first to estimate optical flow with an estimated latency of 10s of μs on asynchronous hardware, which represents a gain of 3 orders of magnitude compared to previous methods. We show that the method can detect abrupt motion changes faster than a periodic output, which may be critical for safety-related applications. This represents an important step towards ultra-low latency and low-power event-based vision.

Acknowledgments

This work was supported by the French National Research Agency (ANR) through the 'Investissement d'avenir' (investments for the future) programs: ANR 10-AIRT-005(IRTNANO-ELEC).

References

[1] https://docs.prophesee.ai/stable/hw/manuals/esp.html, . Accessed: 29-Oct-2024. 4

[2] https://docs.prophesee.ai/stable/api/cpp/cv/algorithms.html, . Accessed: 29-Oct-2024. 4

[3] Ryad Benosman, Charles Clercq, Xavier Lagorce, Sio-Hoi Ieng, and Chiara Bartolozzi. Event-based visual flow. *IEEE Transactions on Neural Networks and Learning Systems*, 25 (2):407–417, 2013. 2

[4] Y Bi, A Chadha, A Abbas, , E Bourtsoulatze, and Y Andreopoulos. Graph-based object classification for neuromorphic vision sensing. In *IEEE/CVF International Conference on Computer Vision (ICCV)*. IEEE, 2019. 2, 3

[5] Thomas Dalgaty, Thomas Mesquida, Damien Joubert, Amos Sironi, Cyrille Soubeyrat, Pascal Vivet, and Christoph Posch. The cnn vs. snn event-camera dichotomy and perspectives for event-graph neural networks. In *2023 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, pages 1–6, 2023. 2

[6] Thomas Dalgaty, Thomas Mesquida, Damien Joubert, Amos Sironi, Pascal Vivet, and Christoph Posch. Hugnet: Hemispherical update graph neural network applied to low-latency event-based optical flow. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, pages 3952–3961, 2023. 2, 3, 4, 5, 6

[7] Yongjian Deng, Hao Chen, Huiying Chen, and Youfu Li. Learning from images: A distillation learning framework for event cameras. *IEEE Transactions on Image Processing*, 30: 4919–4931, 2021. 1, 2

[8] Yongjian Deng, Hao Chen, Hai Liu, and Youfu Li. A voxel graph cnn for object classification with event cameras. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1162–1171, 2022. 2, 3

[9] Ziluo Ding, Rui Zhao, Jiyuan Zhang, Tianxiao Gao, Ruiqin Xiong, Zhaofei Yu, and Tiejun Huang. Spatio-temporal recurrent networks for event-based optical flow estimation. In *AAAI Conference on Artificial Intelligence (AAAI)*, pages 525–533, 2022. 1, 2, 4, 6

[10] Matthias Fey, Jan Eric Lenssen, Frank Weichert, and Heinrich Müller. Splinecnn: Fast geometric deep learning with continuous b-spline kernels. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 869–877, 2018. 3

[11] Guillermo Gallego, Tobi Delbruck, Garrick Orchard, Chiara Bartolozzi, Brian Taba, Andrea Censi, Stefan Leutenegger, Andrew J. Davison, Jorg Conradt, Kostas Daniilidis, and Davide Scaramuzza. Event-Based Vision: A Survey . *IEEE Transactions on Pattern Analysis & Machine Intelligence*, 44 (01):154–180, 2022. 1

[12] Daniel Gehrig and Davide Scaramuzza. Low-latency automotive vision with event cameras. *Nature*, 629:1034–1040, 2024. 1, 2, 3, 8

[13] Mathias Gehrig, Mario Millhäusler, Daniel Gehrig, and Davide Scaramuzza. E-raft: Dense optical flow from event cameras. In *International Conference on 3D Vision (3DV)*, 2021. 1, 2, 4, 5, 6, 8

[14] Kamil Jeziorek, Piotr Wzorek, Krzysztof Blachut, Andrea Pinna, and Tomasz Kryjak. Embedded graph convolutional networks for real-time event data processing on soc fpgas. *arxiv:2406.07318*, 2024. 2

[15] Thomas Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representation (ICLR)*, 2017. 5

[16] Chankyu Lee, Adarsh 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 (ECCV)*, pages 366–382. Springer, 2020. 2, 6

[17] Patrick Lichtsteiner, Christoph Posch, and Tobi Delbruck. A 128× 128 120 db 15 μs latency asynchronous temporal contrast vision sensor. *IEEE Journal of Solid-State Circuits*, 43 (2):566–576, 2008. 1

[18] Min Liu and Tobi Delbruck. Edflow: Event driven optical flow camera with keypoint detection and adaptive block matching. *IEEE Transactions on Circuits and Systems for Video Technology*, 32(9):5776–5789, 2022. 2

[19] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representation (ICLR)*, 2017. 5

[20] Xinglong Luo, Kunming Luo, Ao Luo, Zhengning Wang, Ping Tan, and Shuaicheng Liu. Learning optical flow from event camera with rendered dataset. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 9847–9857, 2023. 1, 2, 5, 6, 7

[21] Xinglong Luo, Ao Luo, Zhengning Wang, Chunyu Lin, Bing Zeng, and Shuaicheng Liu. Efficient meshflow and optical flow estimation from event cameras. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 19198–19207, 2024. 1, 2

[22] Moritz B Milde, Olivier JN Bertrand, Harshawardhan Ramachandran, Martin Egelhaaf, and Elisabetta Chicca. Spiking elementary motion detector in neuromorphic systems. *Neural computation*, 30(9):2384–2417, 2018. 2

[23] Anton Mitrokhin, Zhiyuan Hua, Cornelia Fermüller, and Yiannis Aloimonos. Learning visual motion segmentation using event surfaces. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 14402–14411, 2020. 2, 3

[24] Federico Paredes-Vallés and Guido C. H. E. de Croon. Back to event basics: Self-supervised learning of image reconstruction for event cameras via photometric constancy. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3445–3454, 2021. 1, 2, 6

[25] Sarthak Pathak, Alessandro Moro, Hiromitsu Fujii, Atsushi Yamashita, and Hajime Asama. Virtual reality with motion parallax by dense optical flow-based depth generation

from two spherical images. In *2017 IEEE/SICE International Symposium on System Integration (SII)*, pages 887–892, 2017. 1

[26] Simon Schaefer, Daniel Gehrig, and Davide Scaramuzza. Aegnn: Asynchronous event-based graph neural networks. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. 2, 3

[27] Shintaro Shiba, Yoshimitsu Aoki, and Guillermo Gallego. Fast event-based optical flow estimation by triplet matching. *IEEE Signal Processing Letters*, 29:2712–2716, 2022. 2

[28] Shintaro Shiba, Yoshimitsu Aoki, and Guillermo Gallego. Secrets of event-based optical flow. In *European Conference on Computer Vision (ECCV)*, pages 628–645. Springer, 2022. 2

[29] Timo Stoffregen and Lindsay Kleeman. Event cameras, contrast maximization and reward functions: An analysis. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 12300–12308, 2019. 2

[30] Timo Stoffregen, Cedric Scheerlinck, Davide Scaramuzza, Tom Drummond, Nick Barnes, Lindsay Kleeman, and Robert Mahony. Reducing the sim-to-real gap for event cameras. *European Conference on Computer Vision (ECCV)*, 2020. 1, 2, 6

[31] Yi Tian and Juan Andrade-Cetto. Sdformerflow: Spatiotemporal swin spikeformer for event-based optical flow estimation. *arxiv:2409.04082*, 2024. 2, 6

[32] Daniel Vera-Yanez, António Pereira, Nuno Rodrigues, José Pascual Molina, Arturo S. García, and Antonio Fernández-Caballero. Optical flow-based obstacle detection for mid-air collision avoidance. *Sensors*, 24(10), 2024. 1

[33] Zhexiong Wan, Yuchao Dai, and Yuxin Mao. Learning dense and continuous optical flow from an event camera. *IEEE Transactions on Image Processing*, 31:7237–7251, 2022. 1, 2, 6

[34] Yufeng Yang, Adrian Kneip, and Charlotte Frenkel. Evgnn: An event-driven graph neural network accelerator for edge vision. *arXiv:2404.19489*, 2024. 2, 3, 5

[35] Jiyang Yu and Ravi Ramamoorthi. Learning video stabilization using optical flow. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. 1

[36] Tianwei Zhang, Huayan Zhang, Yang Li, Yoshihiko Nakamura, and Lei Zhang. Flowfusion: Dynamic dense rgb-d slam based on optical flow. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 7322–7328, 2020. 1

[37] Hengshuang Zhao, Li Jiang, Jiaya Jia, Philip Torr, and Vladlen Koltun. Point transformer. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 16239–16248, 2021. 3, 5

[38] Alex Zhu, Liangzhe Yuan, Kenneth Chaney, and Kostas Daniilidis. Ev-flownet: Self-supervised optical flow estimation for event-based cameras. In *Robotics: Science and Systems*, Pittsburgh, Pennsylvania, 2018. 1, 2, 4, 5, 6

[39] A. Z. Zhu, D. Thakur, T. Ozaslan, B. Pfrommer, V. Kumar, and K. Daniilidis. The multi vehicle stereo event camera dataset: An event camera dataset for 3d perception. *IEEE Robotics and Automation Letters*, 3(3):2032–2039, 2018. 4