E-GMFlow: Time granularity for transformer architectures in event-based optical flow

Anca Badiu[1]

Supervisor(s): Hesam Araghi [1]**, Nergis Tömen** [1]

[1]**EEMCS, Delft University of Technology, The Netherlands**

A Thesis Submitted to EEMCS Faculty Delft University of Technology,
In Partial Fulfilment of the Requirements
For the Bachelor of Computer Science and Engineering
June 23, 2024

Name of the student: Anca Badiu
Final project course: CSE3000 Research Project
Thesis committee: Hesam Araghi, Nergis Tömen, Guohao Lan

Abstract

Event cameras are bio-inspired sensors with high dynamic range, high temporal resolution, and low power consumption. These features enable precise motion detection even in challenging lighting conditions and fast-changing scenes, rendering them well-suited for optical flow estimation. However, event camera output is sparse and unstructured, making it challenging to process. Transformer architectures have shown to be effective in capturing long-term temporal dependencies and processing sparse input, hence they might be better suited to processing this output by leveraging the fine time granularity inherent to event camera data. We introduce E-GMFlow, an approach for event-based optical flow inspired by the recent success in terms of accuracy of transformer-based models for frame-based optical flow. We explore the effect of temporal details on the accuracy of this transformer architecture by changing the number of temporal bins in which events are discretized. We observe that the increase in the number of temporal bins generally causes higher accuracy and comment on the limitations of this study.

1 Introduction

Optical flow estimation is a fundamental task in computer vision, for motion analysis of objects in a scene. It is formulated as estimating per-pixel correspondences between two consecutive video frames, in the form of a 2D displacement field. Optical flow estimation is pivotal in various applications, including object tracking [1], video stabilization [2], and video restoration [3]. Event cameras, a novel type of vision sensor, can offer significant improvements in accuracy, latency, and power consumption for optical flow estimation compared to standard cameras. These advantages stem from their unique characteristics: high temporal resolution ($>$ 10K fps), high dynamic range (140 dB versus 60 dB), low power consumption ($<$ 10 mW) and high pixel bandwidth [4]. Given the well-established field of optical flow estimation with standard cameras, it is advantageous to adapt frame-based methods to event cameras to leverage their unique capabilities.

The output of event cameras is unconventional, making it unclear how to adapt frame-based optical flow approaches. Unlike conventional cameras that capture entire frames at fixed time intervals, event cameras asynchronously capture per-pixel brightness changes, mimicking the functionality of the human eye's retina. This results in a stream of events that are challenging to process because of their sparse and unstructured nature. Since event streams are not structured in a 2D grid, like frames, they need to be pre-processed before being passed through models inspired by frame-based approaches such as convolutional neural networks (CNN). Thus, several event representations have been proposed to convert streams to a frame-like structure [5; 6].

Transformer-based architectures are achieving state-of-the-art (SOTA) performance for standard camera optical flow estimation [7; 8], but fewer such architectures are yet used for event-based optical flow. Given their ability to capture long-term dependencies, transformers could improve the processing of the sparse output of event cameras. This leads to the question, **does a higher time granularity in the event representation for a transformer-based model improve accuracy for event-based optical flow?** We measure accuracy using three metrics: N-Point Error (NPE), Angular Error (AE) and End-Point-Error (EPE).

In this paper, we will explore event representations suited for a transformer-based approach inspired by standard cameras and the role that time discretization plays in such a representation. We will be focusing on adapting GMFlow [7], due to its high accuracy for standard camera optical flow estimation and the availability and usability of its code. We use a voxel-grid event representation [9] and compare the accuracy of the model between different numbers of bins for discretizing the time domain.

The paper is organized as follows: Section 2 will discuss the background and related literature; Section 3 provides some background and notation for the concepts used in the rest of the paper; Section 4 details the proposed algorithm as well as a more in-depth explanation of the standard camera approach it is based upon; The experimental setup along with the results of the experiments are shown in Section 5; Section 6 contains a reflection upon the ethical implications of this work and its reproducibility; A discussion and analysis of the results can be found in Section 7 and Section 8 concludes and proposes possible avenues for future work.

2 Related work

Traditionally, for standard cameras, optical flow methods were treated as an energy minimization problem. In this case, image brightness constancy and spatial smoothness between frames are assumed, resulting in a trade-off between a data term enforcing brightness constancy and a regularization term promoting spatial smoothness [10]. In recent years, deep learning techniques have made significant strides in computer vision, including the realm of optical flow estimation, especially methods using convolutional neural networks (CNN) [11; 12; 13]. Yet, CNN-based models cannot capture long-range dependencies, primarily because convolution operations prioritize local information. Methods have been proposed to account for this such as iterative refinement [14] or coarse-to-fine pyramids [15]. Transformer-based architectures are more suited to capturing long-range dependencies and have recently garnered attention, achieving state-of-the-art performance [7; 8; 16; 17].

Frame-based optical flow transformer architectures: GMA [18] was among the first works to incorporate a transformer into optical flow methods, to better account for occluded areas. It uses self-attention to propagate motion features from non-occluded areas to occluded areas and estimate optical flow for these areas. GMFlow [7] reformulates optical flow as a global matching problem to address the challenge of large displacements. To account for mutual relationships between frames it uses a transformer architecture for feature enhancement after constructing a 4D-correlation volume.

Event-based optical flow transformer architectures: Recently, some methods using transformer designs have been proposed for event-based architectures as well. For example, E-FlowFormer [17] proposes a method based on E-RAFT [6] using a transformer design to enhance the event feature encoding, before constructing a 4D-correlation volume.

Event representation for transformer architectures: Due to the unstructured nature of an event stream, it is not immediately clear how to best represent events to exploit the advantages offered by a transformer-based model in the context of optical flow. Common approaches to representing events for various computer vision tasks include frame-based approaches [19], voxel-based approaches [9] or spike-based approaches [20]. A relatively common voxel-based approach to representing events in both transformer designs [17; 21] and Artificial Neural Networks (ANN) [6] is a volumetric voxel grid representation [9]. This works by discretizing the time domain of a batch of events and using temporal bilinear interpolation to improve the resolution.

Some approaches have also been proposed to take advantage of the high temporal resolution of event cameras. For example, a novel transformer-based encoder that directly processes an event sequence without accumulating events in a 2D or 3D space is proposed in [22] and obtains SOTA results for classification tasks. Similarly, Peng et al. [23] first group events asynchronously based on their timestamps and polarities, which are then passed through a novel self-attention mechanism and two aggregation modules to generate event features.

3 Background

In this section, we will provide some necessary background into the working principles of event cameras, CNNs, and transformers.

Event camera: Event-based cameras track the changes in log intensity at each pixel and only generate events when this change exceeds a certain threshold. Each event generated can be understood as a 4-tuple containing the location of the pixel, the timestamp, and the polarity of the brightness change (positive change or negative change):

$$e = (x, y, t, p) \tag{1}$$

This results in a stream of events $E = \{e_i | i \in [1, N]\}$.

CNN: A convolutional neural network is a type of deep learning model that is effective at processing grid-like data, such as images. It utilizes convolutional layers to extract and learn features hierarchically.

Transformer: Transformers are a type of deep learning model introduced by Vaswani et al. [24] centered on the mechanism of self-attention, designed to process sequential data.

4 Proposed method

In this section, we propose an approach that uses the voxel grid representation to adapt GMFlow [7] for an event camera input stream.

4.1 Event representation

As explored in Section 2, there are multiple ways of encoding events and it is not immediately clear which would be preferable. Due to its flexibility in choosing the amount of time information retained, we have decided to focus on the commonly used voxel grid approach [9] that has proved successful in other works that use a CNN feature extractor.

Given a batch of N events $\{e_i | i \in [1, N]\}$, this representation discretizes the time domain into B bins. A simple approach to creating a voxel grid from an event stream would be to add up the polarities of the events inside each voxel. However, this representation treats all events inside one voxel as equally relevant. To improve the amount of encoded time information beyond the number of bins, events are inserted into this volume using the bilinear interpolation function:

$$t_i^* = (B-1)(t_i - t_1)/(t_N - t_1) \tag{2}$$

$$V(x, y, t) = \sum_i p_i \cdot k_b(x - x_i) \cdot k_b(y - y_i) \cdot k_b(t - t_i^*) \tag{3}$$

where $k_b(a) = max(0, 1 - |a|)$ is the bilinear sampling kernel. With this kernel, we make sure that for computing the value at a voxel $V(x, y, t)$, only the polarities of events that are inside the $2 \times 2 \times 2$ space around the point (x, y, t) are taken into account. The polarities are summed up, weighed by how close they are to this point, making events with a timestamp closer to t more relevant for voxel $V(x, y, t)$ than events that are temporally farther away.

Hence, for a batch of N events considering B bins, we would get a volume of size $H \times W \times B$ using this representation, where H and W are the height and width of the image. This can be visualized in Figure 1.

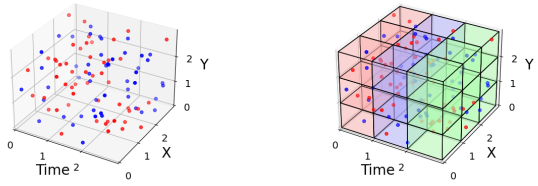

Figure 1: Visual representation of the event stream before (left) and after (right) converting it to a 3x3x3 volumetric voxel grid.

4.2 Original GMFlow algorithm

GMFlow [7] is a frame-based optical flow algorithm with a global matching approach that is effective at dealing with large displacements.

For computing the optical flow between two frames I_1 and I_2 of size $H \times W$, first, each of the frames gets passed through a CNN to extract relevant features downsampled by 1/8. This results in two temporary feature vectors $F_1 \in \mathbb{R}^{H \times W \times D}$ and $F_2 \in \mathbb{R}^{H \times W \times D}$.

The CNN layer can only extract individual features of each frame. Hence, for enhancing this feature extraction process, a

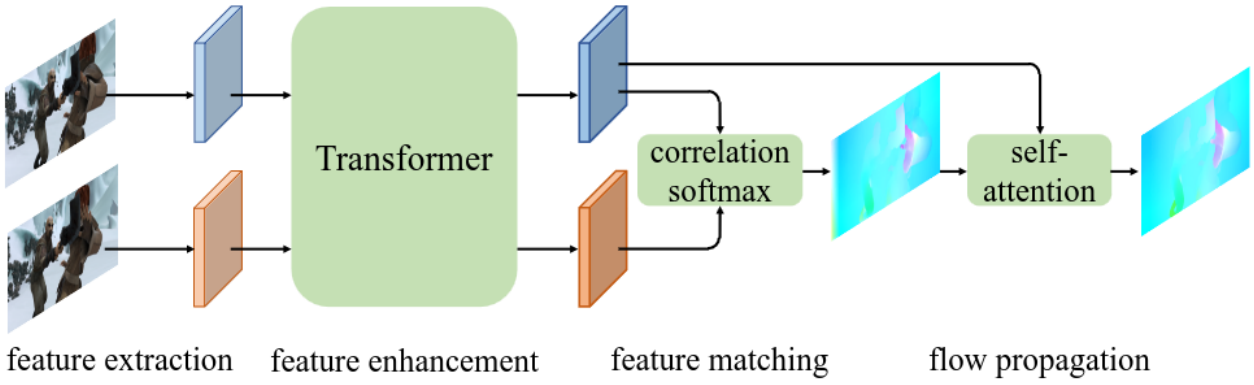

Figure 2: Overview of the GMFlow framework [7]

transformer architecture is used to account for the mutual relationships between the already extracted features F_1 and F_2. To account for the spatial position of each pixel, 2D sine and cosine positional encodings are added. This is followed by six stacked self-attention, cross-attention, and feed-forward network layers, resulting in two feature vectors of length D for each pixel: $\hat{F}_1$ and $\hat{F}_2$.

Similar to E-RAFT [6], a 4D-correlation volume is constructed out of these feature vectors:

$$C = \frac{\hat{F}_1 \hat{F}_2^{\,T}}{\sqrt{D}} \in \mathbb{R}^{H \times W \times H \times W} \tag{4}$$

where the element C_{ijkl} represents the correlation coefficient between coordinates (i, j) in $\hat{F}_1$ and (k, l) in $\hat{F}_2$. Now, for each pixel in I_1, we would like to find a matching pixel in I_2 such the pair has a high correlation value. To do this in a differentiable manner, Xu et al. [7], use a softmax layer on the correlation volume:

$$M = softmax(C) \in \mathbb{R}^{H \times W \times H \times W} \tag{5}$$

To get a flow prediction for a certain pixel (i, j), the weighted average coordinates of the 2D map M_{ij} from that pixel are computed. Finally, to better account for occluded pixels, GMFlow propagates flow predictions from matched pixels to unmatched pixels using a self-attention layer. For refinement, Xu et al. [7] propose a further improvement to this framework by introducing a higher resolution feature of 1/4.

An overview of the architecture of GMFlow can be seen in Figure 2.

4.3 E-GMFlow

To compute the optical flow from time t_i to time t_{i+1}, we choose two short consecutive event sequences from time (t_{i-1}, t_i) and (t_i, t_{i+1}) and process each of these batches of events into a B-dimensional voxel grid (with B bins) as described in Section 4.1.

These two processed event batches get further sent to a slightly modified version of the original GMFlow algorithm. We keep the structure of the algorithm as described in Section 4.2 using the feature matching, flow propagation, and refinement layers. Since the GMFlow is a frame-based approach, its CNN encoder was initially only capable of processing images with 3 channels. In order to be able to test our model with multiple time bin sizes, we adapted the first layer of the CNN feature extractor to be capable of processing more than 3 channels.

5 Experimental Setup and Results

This section presents the experiments conducted to evaluate the accuracy of the proposed optical flow estimation model and their results.

Datasets Two datasets are relevant to our experimental setup: DSEC [25] and KITTI [26]. DSEC is a dataset for event-based vision containing 53 driving sequences captured at various times of day with optical flow ground truth data. Compared to MVSEC [27], another commonly used dataset for event-based optical flow estimation, DSEC provides scenes with larger pixel displacements (up to 210 pixels) and 3 times higher camera resolution. KITTI is a dataset for mobile robotics and autonomous driving research containing 6 hours of traffic scenarios captured with a variety of sensor modalities which can be used for standard camera optical flow prediction.

Training and Testing Due to time constraints and the computational resources required to train a transformer-based model, a pre-trained model was used and fine-tuned. The model was fine-tuned on the entirety of the training split of the DSEC dataset and tested on the testing split of the DSEC dataset. The GMFlow pre-trained model with refinement trained on the KITTI dataset was chosen as a starting point, due to the fact that both the KITTI and DSEC datasets are made up of driving sequences. We fixed the trained weights for the transformer layers and only fine-tuned the weights required for the CNN encoder. The model was fine-tuned for only 7200 iterations. Note that, due to time constraints, the number of iterations used is significantly lower compared to the training of the original GMFlow algorithm which used 100000 iterations. The batch size was also reduced to 6 from the original 16 to fit in GPU memory. For this process, two V100 GPU cards with 32GB were used, which made the

training take up approximately 24 hours per model.

Metrics To measure the performance, we compute three metrics: N-Point Error (NPE) for $N = 1, 2$ or 3, Angular Error (AE), and End-Point Error (EPE). The N-Point Error measures the percentage of pixels with an optical flow magnitude error larger than N. The Angular Error between a flow vector (u, v) and the ground truth flow (u_{GT}, v_{GT}) represents the angle in space between $(u, v, 1)$ and $(u_{GT}, v_{GT}, 1)$ and can be computed with the formula:

$$AE = cos^{-1}\Big(\frac{1 + u \times u_{GT} + v \times v_{GT}}{\sqrt{1 + u^2 + v^2}\sqrt{1.0 + u_{GT}^2 + v_{GT}^2}}\Big) \quad (6)$$

The End-Point Error is simply the L2 norm of the error of the optical flow prediction:

$$EPE = \sqrt{(u - u_{GT})^2 + (v - v_{GT})^2} \quad (7)$$

For all three metrics, a lower value represents higher accuracy.

Experimental setup The model was fine-tuned and tested for 5 bin sizes (3, 5, 10, 15, 20, 25) and compared with the baseline E-Flowformer.

5.1 Convergence

The EPE calculated after every 100 iterations during training can be seen in Figure 3. It can be seen that, although the EPE fluctuates during training, it generally decreases. However, the difference between consecutive evaluations is still relatively large at the moment the training of the model is stopped, suggesting the EPE might continue to significantly decrease if the model is trained longer.

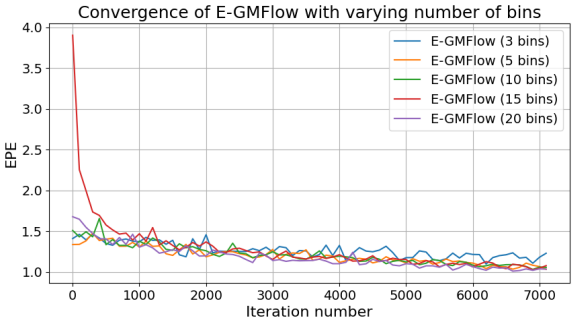

Figure 3: Training EPE for E-GMFlow with 3, 5, 10, 15 and 20 bins. The EPE is calculated during training every 100 iterations on a batch size of 6 samples and averaged.

5.2 Accuracy comparison

Excluding E-GMFlow with 3 time bins, increasing the number of bins causes an increase in all accuracy metrics that we considered. E-GMFlow with 3 time bins performs slightly better than E-GMFlow with 5 bins, but worse than the rest of the models with a higher number of time bins in all metrics. Table 1 shows a comparison between E-GMflow with 3, 5, 10, 15, and 20 bin sizes and the baseline E-FlowFormer. Our model performs worse than the baseline on all metrics, which is to be expected given the small amount of training iterations.

6 Responsible Research

This section will discuss the ethical implications of our work focusing on the reproducibility of the proposed method and the quality of the data used.

6.1 Reproducibility

Reproducibility is a key part of scientific integrity, ensuring that research findings are credible, reliable, and verifiable by others in the field. To ensure the reproducibility of the experiments run in this work, the code used has been made available[1], including testing and training scripts. Both the DSEC and KITTI datasets used during the experiments are publicly available. The method and experimental setup have also been detailed in the paper, ensuring that other researchers can accurately replicate this work.

6.2 Data

An essential component for efficient, ethical, and reproducible research is high-quality data and proper data management. We will discuss how our research adheres to the FAIR (Findable, Accessible, Interoperable, Reusable) principles [28] for scientific data management:

- **Findability:** Both datasets used during the experiments are publicly available.
- **Accessibility:** The KITTI dataset and the training split for the DSEC dataset with optical flow ground truth are fully accessible without authorization. The ground truth for the test split of the DSEC dataset cannot be directly accessed, however, the optical flow predictions can be benchmarked on the official DSEC website [2].
- **Interoperability:** The data format is well-described and easy to use in both cases.
- **Reusability:** The structure of the data is well-explained in both datasets making them easy to reuse for other research purposes.

7 Discussion and Limitations

The experiments were performed on only one specific type of transformer model making it unclear whether the effect is caused by the use of this architecture and would thus generalize to other transformer-based architectures.

The small amount of iterations used for training also puts into question the validity of the results. Since the models have not fully converged, the difference in accuracy between the proposed models using various bin sizes may indicate that transformer-based models using more time bins simply converge faster than models with fewer time bins and do not necessarily perform better in terms of accuracy.

The model was trained and tested on only one dataset (DSEC) consisting of driving scenarios, hence the findings might not generalize for a different type of input data.

[1] https://github.com/ancabd2/rp

[2] https://dsec.ifi.uzh.ch

	1PE	2PE	3PE	AE	EPE
E-Flowformer	11.23	4.10	2.45	2.68	0.76
E-GMflow (3 bins)*	37.21	14.14	7.80	5.27	1.49
E-GMflow (5 bins)*	39.25	15.80	8.70	5.57	1.57
E-GMflow (10 bins)*	34.33	12.70	6.92	5.18	1.38
E-GMflow (15 bins)*	31.32	11.43	6.41	4.89	1.31
E-GMflow (20 bins)*	29.28	10.68	6.03	4.76	1.26

Table 1: Accuracy comparison between E-GMFlow with 5 different numbers of time bins (3, 5, 10, 15 and 20). Here we show the N-PE (for $N = 1$, 2, and 3), AE, and EPE on the DSEC testing split. Algorithms marked with * represent our proposed models.

8 Conclusions and Future Work

In this paper, we propose E-GMFlow, a transformer model for event-based optical flow estimation. We studied the effect of increasing the number of time bins in its event representation on its accuracy.

The analysis suggests that the increase in the number of time bins for E-GMFlow corresponds to an increase in accuracy. However, due to the low number of iterations, it is not entirely clear whether this effect would still be observable if the model was trained for a longer amount of time.

Given the quality of the results for such a low number of iterations, it is likely that E-GMFlow would perform significantly better if trained for a longer time. Future research could investigate this potential by training the model for a longer duration. Since in this work, we focused on using one specific type of event representation, it would be interesting to explore the effect on the performance of other event representation strategies as well. This research has been performed on one specific dataset, but it would be interesting to see how this new method performs in different scenarios.

References

[1] Kiran Kale, Sushant Pawar, and Pravin Dhulekar. Moving object tracking using optical flow and motion vector estimation. In *2015 4th international conference on reliability, infocom technologies and optimization (ICRITO)(trends and future directions)*, pages 1–6. IEEE, 2015.

[2] Anli Lim, Bharath Ramesh, Yue Yang, Cheng Xiang, Zhi Gao, and Feng Lin. Real-time optical flow-based video stabilization for unmanned aerial vehicles. *Journal of Real-Time Image Processing*, 16:1975–1985, 2019.

[3] Kelvin CK Chan, Xintao Wang, Ke Yu, Chao Dong, and Chen Change Loy. Basicvsr: The search for essential components in video super-resolution and beyond. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4947–4956, 2021.

[4] Guillermo Gallego, Tobi Delbrück, Garrick Orchard, Chiara Bartolozzi, Brian Taba, Andrea Censi, Stefan Leutenegger, Andrew J Davison, Jörg Conradt, Kostas Daniilidis, et al. Event-based vision: A survey. *IEEE transactions on pattern analysis and machine intelligence*, 44(1):154–180, 2020.

[5] 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.

[6] Mathias Gehrig, Mario Millhäusler, Daniel Gehrig, and Davide Scaramuzza. E-raft: Dense optical flow from event cameras. In *2021 International Conference on 3D Vision (3DV)*, pages 197–206. IEEE, 2021.

[7] Haofei Xu, Jing Zhang, Jianfei Cai, Hamid Rezatofighi, and Dacheng Tao. Gmflow: Learning optical flow via global matching. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 8121–8130, 2022.

[8] Zhaoyang Huang, Xiaoyu Shi, Chao Zhang, Qiang Wang, Ka Chun Cheung, Hongwei Qin, Jifeng Dai, and Hongsheng Li. Flowformer: A transformer architecture for optical flow. In *European conference on computer vision*, pages 668–685. Springer, 2022.

[9] 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.

[10] Berthold KP Horn and Brian G Schunck. Determining optical flow. *Artificial intelligence*, 17(1-3):185–203, 1981.

[11] Deqing Sun, Xiaodong Yang, Ming-Yu Liu, and Jan Kautz. Pwc-net: Cnns for optical flow using pyramid, warping, and cost volume. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 8934–8943, 2018.

[12] Alexey Dosovitskiy, Philipp Fischer, Eddy Ilg, Philip Hausser, Caner Hazirbas, Vladimir Golkov, Patrick Van Der Smagt, Daniel Cremers, and Thomas Brox. Flownet: Learning optical flow with convolutional networks. In *Proceedings of the IEEE international conference on computer vision*, pages 2758–2766, 2015.

[13] Zachary Teed and Jia Deng. Raft: Recurrent all-pairs field transforms for optical flow. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part II 16*, pages 402–419. Springer, 2020.

[14] Junhwa Hur and Stefan Roth. Iterative residual refinement for joint optical flow and occlusion estimation. In

Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pages 5754–5763, 2019.

[15] Anurag Ranjan and Michael J Black. Optical flow estimation using a spatial pyramid network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4161–4170, 2017.

[16] Andrew Jaegle, Sebastian Borgeaud, Jean-Baptiste Alayrac, Carl Doersch, Catalin Ionescu, David Ding, Skanda Koppula, Daniel Zoran, Andrew Brock, Evan Shelhamer, et al. Perceiver io: A general architecture for structured inputs & outputs. *arXiv preprint arXiv:2107.14795*, 2021.

[17] Yijin Li, Zhaoyang Huang, Shuo Chen, Xiaoyu Shi, Hongsheng Li, Hujun Bao, Zhaopeng Cui, and Guofeng Zhang. Blinkflow: A dataset to push the limits of event-based optical flow estimation. In *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3881–3888. IEEE, 2023.

[18] Shihao Jiang, Dylan Campbell, Yao Lu, Hongdong Li, and Richard Hartley. Learning to estimate hidden motions with global motion aggregation. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 9772–9781, 2021.

[19] 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.

[20] Jun Haeng Lee, Tobi Delbruck, and Michael Pfeiffer. Training deep spiking neural networks using backpropagation. *Frontiers in neuroscience*, 10:228000, 2016.

[21] Wenming Weng, Yueyi Zhang, and Zhiwei Xiong. Event-based video reconstruction using transformer. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2563–2572, 2021.

[22] Zhihao Li, M Salman Asif, and Zhan Ma. Event transformer. *arXiv preprint arXiv:2204.05172*, 2022.

[23] Yansong Peng, Yueyi Zhang, Zhiwei Xiong, Xiaoyan Sun, and Feng Wu. Get: group event transformer for event-based vision. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6038–6048, 2023.

[24] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[25] Mathias Gehrig, Willem Aarents, Daniel Gehrig, and Davide Scaramuzza. Dsec: A stereo event camera dataset for driving scenarios. *IEEE Robotics and Automation Letters*, 2021.

[26] Andreas Geiger, Philip Lenz, Christoph Stiller, and Raquel Urtasun. Vision meets robotics: The kitti dataset. *The International Journal of Robotics Research*, 32(11):1231–1237, 2013.

[27] 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.

[28] Mark D Wilkinson, Michel Dumontier, IJsbrand Jan Aalbersberg, Gabrielle Appleton, Myles Axton, Arie Baak, Niklas Blomberg, Jan-Willem Boiten, Luiz Bonino da Silva Santos, Philip E Bourne, et al. The fair guiding principles for scientific data management and stewardship. *Scientific data*, 3(1):1–9, 2016.