Asynchronous Collaborative Graph Representation for Frames and Events

Supplementary Material

Table 7. Comparison with state-of-the-art methods on the DSEC-Detection dataset [13] with two classes (i.e., pedestrian and car).

Modality	Method	Representation	mAP	mAP_{50}
Event	RVT [15]	Voxel Grid	0.384	0.587
	SAST [30]	Event Volume	0.381	0.601
	SSM [39]	Voxel Grid	0.380	0.552
Frame+Event	DAGr-18 [13]	Frame+Graph	0.376	-
	DAGr-34 [13]	Frame+Graph	0.390	-
	DAGr-50 [13]	Frame+Graph	0.419	0.660
	DAGr [13]	ACGR	**0.457**	**0.688**
	SODFormer [22]	ACGR	**0.523**	**0.742**

A. More Details on DSEC-Detection

A.1. DSEC-Detection Dataset

DSEC [16] is a stereo camera dataset designed for driving scenarios, comprising data captured by two monochrome event cameras (i.e., Prophesee Gen3.1) and two global shutter color cameras (i.e., FLIR Blackfly S USB3). The dataset encompasses various lighting conditions, including favorable and challenging illumination environments, to support robust evaluation of vision-based systems. As an extension of DSEC, the DSEC-Detection dataset [13] is specifically tailored for object detection tasks. It contains 60 hybrid sequences (53 from DSEC and 7 additional with challenging scenarios), of which 47 are for training and 13 for evaluation. The spatial resolution of the visual streams is 640×480, and the frame rate of RGB frames is 20 Hz. The bounding boxes are generated by Quasi-Dense Tracking based on the RGB frames and manually corrected, with a total of 70,379 frames and 390,118 annotations. The DSEC-Detection dataset involves 8 classes: pedestrian, rider, car, bus, truck, bicycle, motorcycle, and train. ***However, some existing studies [13] evaluate object detection models using only two classes: pedestrian and a consolidated "car" category combining car, bus, and truck.***

A.2. Extensive Comparisons with Two classes

As clarified in Sec. 4.1, we evaluate the ACGR on the full 8 classes in Sec 4. For a fair comparison, we conduct quantitative evaluation experiments on the two-class DSEC-Detection dataset. As shown in Table 7, the results show that with the ACGR, the mAP of DAGr considerably improves by 3.8%. Furthermore, SODFormer achieves satisfactory results leveraging ACGR as representation, reaching an extraordinary mAP of 0.523. Compared to event-based unimodal approaches, our methods exhibit exceptional performance with a distinct advantage of at least 7.3% mAP improvement. In other words, this two-class test further confirms the effectiveness of the proposed ACGR.

B. Event Representations

Numerous methods have been proposed to design event representations to make asynchronous events compatible with deep learning methods. To compare these representations and highlight the advantages of ACGR, we summarize widely used event representations along with their brief descriptions and characteristics in Table 8.

Specifically, we categorize the existing event representations broadly into five types. The first type typically projects events onto a 2D image based on their spatial position and polarity. While these methods are straightforward and efficient, they often discard temporal information. Second, a variety of more complex handcrafted methods have been developed to preserve temporal information. However, these methods are highly sensitive to hyperparameters (e.g., the number of temporal bins in voxel grids) and generally lack generalizability. To this end, some approaches employ ANNs to generate learning-based representations, achieving state-of-the-art performance. However, these approaches come at the cost of increased computational complexity and energy consumption. Recently, emerging brain-inspired SNNs have been proposed to process event streams as binary spikes, offering low-energy solutions. Nevertheless, their performance is currently constrained by the training algorithms, making them unable to match ANNs in performance. Moreover, all four types above face challenges in processing asynchronous event-by-event input, resulting in significant redundant computation during continuous inference. Although SNNs have the potential to overcome this issue, relevant research remains limited.

In contrast, graph-based representations simultaneously retain the spatiotemporal properties and sparsity of event streams while supporting asynchronous input, making them the most effective representation for processing event streams. Moreover, the ACGR is the first trail to introduce RGB frames into the graph representation, significantly enhancing its capacity to adapt to diverse scenarios.

C. Preliminaries

C.1. Spline Convolution

In the GMAL module (see Sec. 3.3), we utilize spline convolution [10] to learn the ACGR by message passing and cross-modal interaction. Here we elaborate on the implementation of spline convolutions for legibility. Technically, spline convolution uses continuous B-splines as kernels for graph convolutions. As formulated in Eq. 7, the edge feature e_{ij} is mapped to the convolutional weight with the

Table 8. Comparison of some representative event representation methods.

Type	Event representation	Dimensions	Description	Temporal	Polarity	Asyn.
Event images	Binary image [31]	$2 \times H \times W$	Two-channel image of event polarities	×	√	×
	Event count image [27]	$2 \times H \times W$	Rate-based image of event counts	×	√	×
	Grayscale image [34]	$H \times W$	Binarized grayscale image after filtering	√	√	×
Handcrafted features	Voxel grid [37]	$B \times H \times W$	Voxel grid summing event polarities	√	×	×
	Voxel cube [7]	$C \times T \times H \times W$	4D voxel grid tensor	√	×	×
	Hyper histogram [29]	$4K \times H \times W$	3D temporal histograms	√	√	×
	TORE volume [1]	$2K \times H \times W$	4D voxel grid of last K timestamps	√	√	×
	SAE [5]	$H \times W$	Time surface of active events	√	√	×
	SSR [28]	$C \times H \times W$	Sparsely updating with each new event	√	×	√
ANN-based	EST [14]	$2 \times B \times H \times W$	Learning an event spike tensor using MLPs	√	√	×
	Matrix-LSTM [4]	$C \times H \times W$	Learning an event tensor using LSTMs	√	√	×
	Dense memory cells [18]	$L \times D$	Event cells using MLPs and the attention	√	√	×
	Event embedding [24]	$C \times H \times W$	Learning an event tensor using TACN	√	√	×
	Reconstructed image [32]	$H \times W$	Intensity image using E2VID algorithm	√	×	×
	SET [17]	$2 \times H \times W$	Learning sparse event tensors using LSTMs	√	√	×
SNN-based	Spike image [23]	$2 \times H \times W$	Two-channel image using the LIF model	√	√	×
	Leaky surface [3]	$H \times W$	Event image using the leak surface layer	√	×	×
	ARSNN [35]	$C \times H \times W$	Event tensor using the recurrent SNNs	√	√	×
Graph-based	Event Graph [2]	$N \times 1$	Using events as nodes and polarity as feature	√	√	√
	Spatio-Temporal Clouds [6]	$N \times 3$	Using events as nodes and position as feature	√	×	√
	Voxel Graph [8]	$N_p \times D$	Voxeling events into nodes	√	√	√
	ACGR	$(N_f + N_e) \times D$	Integrating frames and events into one graph	√	√	√

learnable function $\mathcal{W}'$, which is a smoothed matrix-valued function modeled by a 1-order 2-dimensional B-spline with 5×5 learnable weights $\theta_{\mathcal{W}'}$. More concisely, the learnable weights $\theta_{\mathcal{W}'}$ act as the control points in the B-spline, determining the value of the B-spline at each 2D coordinate in $[0, 1]^2$. For each edge feature e_{ij}, $\mathcal{W}'(e_{ij})$ is the value of the B-spline at e_{ij} interpolated by the basis functions and $\theta_{\mathcal{W}'}$. For more details, please refer to [10].

C.2. Adversarial Domain Adaptation

We provide some preliminaries about Adversarial Domain Adaptation (ADA) to better understand the novel training strategy introduced in Sec. 3.3. Domain adaptation aims to address the challenge of applying a model trained on the source domain to the target domain with a different data distribution by aligning the distributions of the source and target domains. Inspired by Generative Adversarial Networks (GANs), ADA achieves this by leveraging similar learning principles to train a domain-invariant feature extractor and a discriminator. Concretely, ADA introduces a domain discriminator that tries to distinguish between source and target domain features while training a feature extractor to confuse the discriminator. Incorporated with the task-specific head shown in Fig. 4, the overall optimization objective can be expressed as:

$$\min_{F,T} \max_{D} \mathcal{L}_t(F, T) - \lambda \mathcal{L}_d(F, D), \tag{12}$$

where F, T, and D are the feature extractor, the task-specific head, and the discriminator. The gradient reversal layer [11] reverses the gradient direction for F to avoid alternating between minimization and maximization.

Despite the strong capability ADA has demonstrated in domain alignment within transfer learning, its application in multimodal fusion remains underexplored. Intuitively, the features of different modalities reside in distinct feature spaces, making cross-modal interaction less effective since they cannot interpret each other's features. To address this limitation, the proposed multimodal training strategy explicitly aligns the feature spaces of different modalities using ADA before cross-modal interaction. Considering the potential instability introduced by ADA during training, we first pre-train the ACGR without ADA and subsequently integrate ADA to enhance cross-modal interaction.

D. Detailed Network Architecture

For clarity, we specify the detailed network architecture and layer parameters in Table 9. The output sizes and pooling parameters listed in Table 9 are derived under the assumption of an input size of $T = 10, H = 480, W = 640$, where the three dimensions of downsampling rate and output size correspond to T, H, W, respectively. LUT-SC represents Lookup-Table SplineConv [13], which accelerates computations by obtaining $\mathcal{W}'(e_{ij})$ via a lookup table. Conv2D(c_{in}, c_{out}) indicates the input channel is c_{in} and the output channel is c_{out}, and similar conventions apply to Conv3D and LUT-SC layers. The Pooling layer with arguments g_t, g_y, g_x denotes the size of grid cells in each dimension. From Table 9, we can see that the frames and

Table 9. Detailed architecture specifications in ACGR.

Stage	Downsp. rate (output size)	Layer Name	Frame	Event
Graph Construction	1×, 2×, 2× (10×240×320)	Patch / Voxel Embedding	Conv2D(3, 32), BN	Conv3D(2, 32), BN
GMAL	1× (10×240×320)	Layer 1	LUT-SC(35, 32), BN LUT-SC(32, 32), BN	
	10×, 4×, 4× (1×60×80)	Pooling 1	Pooling(10, 4, 4)	
		Layer 2	LUT-SC(35, 64), BN LUT-SC(64, 64), BN	
	1×, 4×, 4× (1×15×20)	Pooling 2	Pooling(1, 4, 4)	
		Layer 3	LUT-SC(67, 128), BN LUT-SC(128, 128), BN	

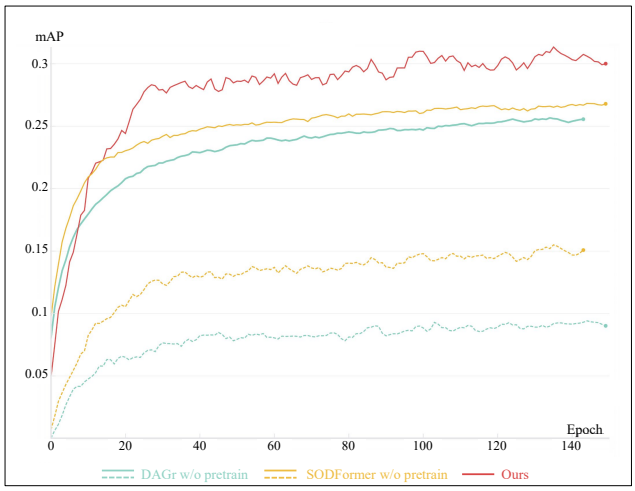

Figure 8. Comparison of the training progress between different methods with or without pre-train on DSEC-Detection [13, 16].

events are integrated into a unified graph before entering the GMAL module, thus reducing the model size and computational complexity. In the GMAL module, the input channel for the first LUT-SC block in each layer is $c_{in} + 3$, indicating that the spatiotemporal coordinates of the nodes are also incorporated as features, which has been verified to enhance performance [13]. Note that most backbones inherently possess the ability to output multiscale features (e.g., DAGr [13], YOLOX [12], and SODFormer [22]), so the part from Pooling 1 to Layer 3 can be optionally omitted as implemented in Sec. 4. This further reduces the model size and improves inference speed.

E. More Experiments

E.1. Comparison of Training Progress

As described in Sec. 4.2, ACGR integrates frames and events at the representation stage, eliminating the need for a two-branch backbone which significantly reduces the model size and training expense. To confirm this, we compare the mAP curves evaluated on the validation set during training for two methods (i.e., DAGr [13] and SODFormer [22]) with and without pretraining, as well as SODFormer using ACGR, as shown in Fig. 8. Notably, both DAGr and SODFormer rely on two-branch backbones to extract features from each modality separately. From Fig. 8, it is evident that these two methods converge very slowly without pretraining (i.e., still improving after 150 epochs) and are inclined to yield sub-optimal convergence. Only by pretraining the unimodal branches as [13] can these two-branch multimodal models fully leverage their potential performance benefits. In contrast, the ACGR requires only a single-branch backbone and achieves multimodal fusion, converging rapidly to the optimum without any pre-train and further enhancing performance through the proposed modules. In summary, our ACGR makes the end-to-end training of multimodal models more efficient.

E.2. Object Detection on PKU-DAVIS-SOD

The PKU-DAVIS-SOD dataset [22] is a multimodal object detection benchmark collected using a DAVIS346 camera, which captures synchronized RGB frames and event streams. Both of them are recorded at 346×260 spatial resolution with an RGB frame rate of 25 Hz. The dataset covers diverse scenarios including normal, low-light, and fast-motion conditions, with precise manually labeled bounding boxes based on RGB frames including three categories (i.e., car, pedestrian and two-wheeler). Consisting of 220 hybrid driving sequences, this dataset contains 276k RGB frames and 1080.1k bounding boxes, being the largest neuromorphic multimodal object detection dataset to date.

To further investigate the generalisability of ACGR on other datasets, we carry out comparisons between our methods and state-of-the-art methods on the PKU-DAVIS-SOD dataset as shown in Table 10. In particular, we compare our methods (i.e., DAGr [13] and SODFormer [22] using ACGR) with ten unimodal detectors, including four frame-based detectors (i.e., Faster R-CNN [33], YOLOv3 [9], Deformable DETR [38], and LSTM-SSD [25]) and four

Table 10. Comparison with state-of-the-art methods on the PKU-DAVIS-SOD dataset [22].

Modality	Method	Representation	Backbone	Temporal	mAP_{50}	Runtime (ms)
Frame	Faster R-CNN [33]	Frame	ResNet50	×	0.443	11.6
	YOLOv3 [9]	Frame	Darknet53	×	0.426	7.9
	Deformable DETR [38]	Frame	Deformable DETR	×	0.461	21.6
	LSTM-SSD [25]	Frame	SSD	√	0.456	22.4
Event	NGA-event [20]	Voxel grid	Darknet53	×	0.232	8.0
	YOLOv3 [9]	Reconstructed image	Darknet53	×	0.244	178.5
	Faster R-CNN [33]	Event image	ResNet50	×	0.251	11.5
	Deformable DETR [38]	Event image	Deformable DETR	×	0.307	21.6
	LSTM-SSD [25]	Event image	SSD	√	0.273	22.7
	ASTMNet [24]	Event embedding	Rec-Conv-SSD	√	0.291	21.3
Frame+Event	MFEPD [21]	Frame+Event image	Darknet53	×	0.438	8.2
	JDF [23]	Frame+Channel image	Darknet54	×	0.442	8.3
	DAGr [13]	Frame+Graph	ResNet50+DAGr-s	√	0.492	18.5
	SODFormer [22]	Frame+Event image	STE+TDTE	√	0.504	39.7
	DAGr [13]	ACGR	DAGr-s	√	0.504	**6.9**
	SODFormer [22]	ACGR	STE+TDTE	√	**0.519**	16.8

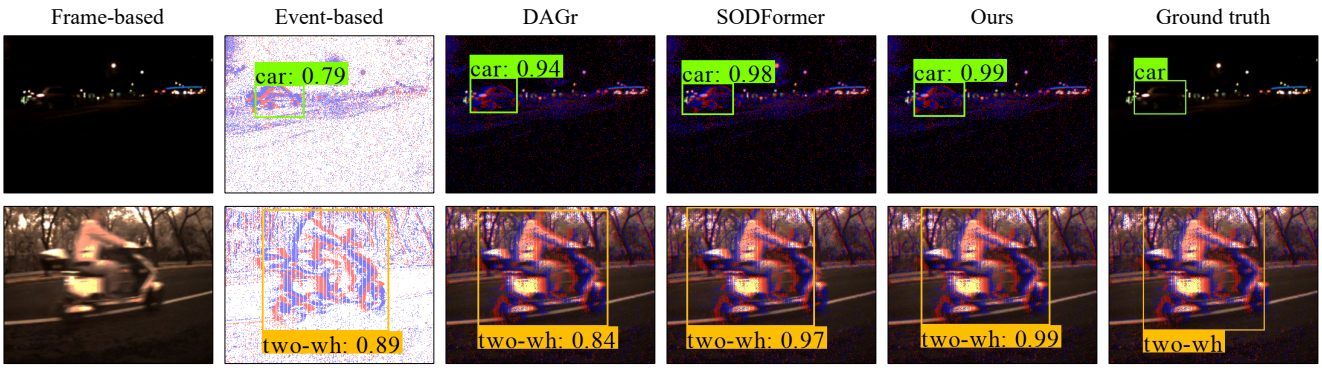

Figure 9. Representative instances of different object detection methods in challenging scenarios on the PKU-DAVIS-SOD dataset [22]. The two rows display the detections in low-light and high-speed motion blur scenarios.

event-based detectors (i.e., NGA [20], reconstructed image [32] for YOLOv3 [9], event image [27] for Faster R-CNN [33], Deformable DETR [38] and LSTM-SSD [25], and ASTMNet [24]), and four state-of-the-art multimodal approaches (i.e., MFEPD [21], JDF [23], DAGr [13], and SODFormer [22]). As presented in Table 10, our methods consistently outperform all the ten unimodal baselines, achieving a mAP_{50} improvement of at least 4.3%, and demonstrate significant advantages over multimodal baselines, with mAP_{50} gains of 1.2% and 1.5% for DAGr and SODFormer, respectively. Additionally, the ACGR reduces inference time by 0.37× on DAGr and 0.42× on SODFormer compared to their baselines, verifying the ability of ACGR to increase efficiency. As illustrated in Fig. 9, we further present a comparison of detections under challenging scenarios (i.e., low light and high-speed motion blur) from the PKU-DAVIS-SOD dataset. These results further demonstrate the reliability of the ACGR.

Table 11. The contribution of each component to our ACGR on the MVSEC dataset [36].

Method	Event	ACAM	$\mathcal{L}_d$	Abs.Rel.↓	RMS↓	RMSlog↓
Baseline				0.306	7.921	0.378
(a)	√			0.279	7.677	0.363
(b)	√	√		0.270	7.536	0.349
Ours	√	√	√	**0.256**	**7.113**	**0.331**

E.3. Contribution Validation on Depth Estimation

To further validate the effectiveness of the proposed modules beyond the object detection task, we adopt the same experimental setup as in the first paragraph of Sec. 4.4 and compare the performance of the four methods on the depth estimation task using the MVSEC dataset [36]. As shown in Table 11, methods (a), (b), and our ACGR also show progressive improvements on the MVSEC dataset, confirming the effectiveness of our designs on the depth estimation

Table 12. Comparison between processing frames using CNNs and ACGR. The backbone utilized here is YOLOX [12].

Method	mAP	mAP_{50}	Params	Runtime (ms)
ResNet18	0.264	0.476	3.41M	16.6
ResNet50	0.305	0.511	9.25M	25.7
ACGR	**0.313**	**0.513**	**0.34M**	**13.2**

task. More specifically, compared to the baseline, methods (a), (b), and our ACGR reduce the Abs.Rel. metric by 0.027, 0.036, and 0.050, with similar improvements observed in the RMS and RMSlog metrics. In other words, it indicates that our ACGR is an adaptive representation suitable for multi-tasks.

E.4. Why Using Graph for Frames.

Although the benefits of ACGR have been validated through various experiments, some might question the reasoning behind representing frames as graphs, especially since frame operations resemble those in CNNs, as discussed in Sec. 3.1. To address this, we outline three key reasons for choosing graphs as the representation for frames.

First, there has been some research on processing frames using GNNs. Vision GNN [19] represents an image as a graph of patches, aggregating contextual information more flexibly and achieving state-of-the-art performance on image recognition and object detection tasks. CoCs [26] treats an image as unorganized points and uses a simplified clustering algorithm to extract features with notable interpretability. While research in this direction is still limited, it holds considerable potential for future advancements.

Second, representing frames as graphs enables more effective cross-modal interaction with events. Indeed, DAGr adopts a paradigm where CNNs process frames and GNNs handle events, yet allowing only unidirectional cross-modal interactions from frames to events, which is likely due to the absence of efficient sampling algorithms. Moreover, simple concatenation of frame and event features leads to inadequate interaction and reduced robustness. In contrast, the ACGR employs a GNN to learn adaptive weights for cross-modal interaction which enhances feature extraction.

Finally, we use the first three layers of ResNet18 and ResNet50 to replace the corresponding layers to extract frame features and compare the results with ACGR in Table 12. The results show that ACGR achieves a significant 4.9% mAP improvement compared to ResNet18, demonstrating its superior capability in extracting frame features. When compared to ResNet50, ACGR maintains a 0.8% mAP advantage due to its mutual interaction which suggests that its feature extraction ability is comparable to ResNet50, while significantly reducing model size and inference time. Overall, ACGR outperforms the CNN+GNN approach in terms of both performance and efficiency.

References

[1] R Wes Baldwin, Ruixu Liu, Mohammed Almatrafi, Vijayan Asari, and Keigo Hirakawa. Time-ordered recent event (tore) volumes for event cameras. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(2):2519–2532, 2022. 2

[2] Yin Bi, Aaron Chadha, Alhabib Abbas, Eirina Bourtsoulatze, and Yiannis Andreopoulos. Graph-based spatio-temporal feature learning for neuromorphic vision sensing. *IEEE Transactions on Image Processing*, 29:9084–9098, 2020. 2

[3] 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

[4] Marco Cannici, Marco Ciccone, Andrea Romanoni, and Matteo Matteucci. A differentiable recurrent surface for asynchronous event-based data. In *Proceedings of the European Conference on Computer Vision*, pages 136–152, 2020. 2

[5] Guang Chen, Hu Cao, Canbo Ye, Zhenyan Zhang, Xingbo Liu, Xuhui Mo, Zhongnan Qu, Jörg Conradt, Florian Röhrbein, and Alois Knoll. Multi-cue event information fusion for pedestrian detection with neuromorphic vision sensors. *Frontiers in Neurorobotics*, 13:10, 2019. 2

[6] Junming Chen, Jingjing Meng, Xinchao Wang, and Junsong Yuan. Dynamic graph cnn for event-camera based gesture recognition. In *Proceedings of the IEEE International Symposium on Circuits and Systems*, pages 1–5, 2020. 2

[7] Loïc Cordone, Benoît Miramond, and Philippe Thierion. Object detection with spiking neural networks on automotive event data. In *Proceedings of the International Joint Conference on Neural Networks*, pages 1–8, 2022. 2

[8] Yongjian Deng, Hao Chen, Hai Liu, and Youfu Li. A voxel graph cnn for object classification with event cameras. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1172–1181, 2022. 2

[9] Ali Farhadi and Joseph Redmon. Yolov3: An incremental improvement. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1–6, 2018. 3, 4

[10] Matthias Fey, Jan Eric Lenssen, Frank Weichert, and Heinrich Müller. Splinecnn: Fast geometric deep learning with continuous b-spline kernels. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 869–877, 2018. 1, 2

[11] Yaroslav Ganin and Victor Lempitsky. Unsupervised domain adaptation by backpropagation. In *Proceedings of the International Conference on Machine Learning*, pages 1180–1189, 2015. 2

[12] Z Ge. Yolox: Exceeding yolo series in 2021. *arXiv*, 2021. 3, 5

[13] Daniel Gehrig and Davide Scaramuzza. Low-latency automotive vision with event cameras. *Nature*, 629(8014):1034–1040, 2024. 1, 2, 3, 4

[14] Daniel Gehrig, Antonio Loquercio, Konstantinos G Derpanis, and Davide Scaramuzza. End-to-end learning of repre-

sentations for asynchronous event-based data. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5633–5643, 2019. 2

[15] Mathias Gehrig and Davide Scaramuzza. Recurrent vision transformers for object detection with event cameras. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13884–13893, 2023. 1

[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. 1, 3

[17] Zhaoxuan Guo, Jiandong Gao, Guangyuan Ma, and Jiangtao Xu. Spatio-temporal aggregation transformer for object detection with neuromorphic vision sensors. *IEEE Sensors Journal*, 2024. 2

[18] Ryuhei Hamaguchi, Yasutaka Furukawa, Masaki Onishi, and Ken Sakurada. Hierarchical neural memory network for low latency event processing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 22867–22876, 2023. 2

[19] Kai Han, Yunhe Wang, Jianyuan Guo, Yehui Tang, and Enhua Wu. Vision gnn: An image is worth graph of nodes. *Advances in Neural Information processing Systems*, 35:8291–8303, 2022. 5

[20] Yuhuang Hu, Tobi Delbruck, and Shih-Chii Liu. Learning to exploit multiple vision modalities by using grafted networks. In *Proceedings of the European Conference on Computer Vision*, pages 85–101, 2020. 4

[21] Zhuangyi Jiang, Pengfei Xia, Kai Huang, Walter Stechele, Guang Chen, Zhenshan Bing, and Alois Knoll. Mixed frame-/event-driven fast pedestrian detection. In *International Conference on Robotics and Automation*, pages 8332–8338, 2019. 4

[22] Dianze Li, Yonghong Tian, and Jianing Li. Sodformer: Streaming object detection with transformer using events and frames. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(11):14020–14037, 2023. 1, 3, 4

[23] Jianing Li, Siwei Dong, Zhaofei Yu, Yonghong Tian, and Tiejun Huang. Event-based vision enhanced: A joint detection framework in autonomous driving. In *Proceedings of the IEEE International Conference on Multimedia and Expo*, pages 1396–1401, 2019. 2, 4

[24] Jianing Li, Jia Li, Lin Zhu, Xijie Xiang, Tiejun Huang, and Yonghong Tian. Asynchronous spatio-temporal memory network for continuous event-based object detection. *IEEE Transactions on Image Processing*, 31:2975–2987, 2022. 2, 4

[25] Mason Liu and Menglong Zhu. Mobile video object detection with temporally-aware feature maps. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5686–5695, 2018. 3, 4

[26] Xu Ma, Yuqian Zhou, Huan Wang, Can Qin, Bin Sun, Chang Liu, and Yun Fu. Image as set of points. In *International Conference on Learning Representations*, 2023. 5

[27] 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. 2, 4

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

[29] Yansong Peng, Yueyi Zhang, Peilin Xiao, Xiaoyan Sun, and Feng Wu. Better and faster: Adaptive event conversion for event-based object detection. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 2056–2064, 2023. 2

[30] Yansong Peng, Hebei Li, Yueyi Zhang, Xiaoyan Sun, and Feng Wu. Scene adaptive sparse transformer for event-based object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16794–16804, 2024. 1

[31] Henri Rebecq, Timo Horstschaefer, and Davide Scaramuzza. Real-time visual-inertial odometry for event cameras using keyframe-based nonlinear optimization. In *Proceedings of the British Machine Vision Conference (BMVC)*, pages 16–1, 2017. 2

[32] Henri Rebecq, René Ranftl, Vladlen Koltun, and Davide Scaramuzza. Events-to-video: Bringing modern computer vision to event cameras. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3857–3866, 2019. 2, 4

[33] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(6):1137–1149, 2016. 3, 4

[34] Yuan-Kai Wang, Shao-En Wang, and Ping-Hsien Wu. Spike-event object detection for neuromorphic vision. *IEEE Access*, 11:5215–5230, 2023. 2

[35] Ziming Wang, Ziling Wang, Huaning Li, Lang Qin, Runhao Jiang, De Ma, and Huajin Tang. Eas-snn: End-to-end adaptive sampling and representation for event-based detection with recurrent spiking neural networks. *arXiv preprint arXiv:2403.12574*, 2024. 2

[36] 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. 4

[37] 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

[38] Xizhou Zhu, Weijie Su, Lewei Lu, Bin Li, Xiaogang Wang, and Jifeng Dai. Deformable DETR: deformable transformers for end-to-end object detection. In *Proceedings of the International Conference on Learning Representations*, 2021. 3, 4

[39] Nikola Zubic, Mathias Gehrig, and Davide Scaramuzza. State space models for event cameras. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5819–5828, 2024. 1