Event-based Graph Representation with Spatial and Motion Vectors for Asynchronous Object Detection

Supplementary Material

1. Training Details

This section outlines the training methodology for the detection task on *eGSMV*. To complement the main paper, we provide additional details on the loss functions and data augmentation techniques used to optimize the graph-based data structure during training.

1.1. Loss Modeling

Here, we discuss the loss modeling associated with our event-driven detection head, which is inspired by YOLOX [5]. The detection head comprises a classification branch and a regression branch, each with a distinct objective and corresponding loss function.

1.1.1. Classification branch

The objective of the classification branch is to classify the object class of each node. Given the inherent class imbalance in the datasets and the varying scale of object classes, the number of nodes corresponding to each object class can differ significantly. To address this, a weighted cross-entropy loss is computed for each node, defined as,

$$l_{cls} = -\frac{1}{N}\sum_{i=1}^{N} w_{y_i} \cdot y_i \cdot \log(\hat{y}_i) \tag{1}$$

Here, y_i is the predicted class probability, $\hat{y}_i$ is the one-hot vector of the ground truth class, and w_{y_i} is a constant weight assigned to the ground truth class of node v_i to account for the class imbalance in the dataset.

1.1.2. Regression Branch

The regression branch is responsible for predicting the relative bounding box coordinates and the Intersection over Union (IoU) confidence, s. Losses are only considered if the node is a non-background event. To improve localization accuracy and penalize incorrect bounding box dimensions, we use cIoU loss [17] and Huber loss [8], computed as follows,

$$l_{\text{loc}} = \frac{1}{\sum_{i\in\mathbb{N}} \mathbf{1}_{\{c_i \neq \text{bg}\}}} \sum_{i\in\mathbb{N}} \mathbf{1}_{\{c_i \neq \text{bg}\}} \cdot \ell_{ciou}(x_i, \hat{x}_i) \tag{2}$$

$$l_{\text{dim}} = \frac{1}{\sum_{i\in\mathbb{N}} \mathbf{1}_{\{c_i \neq \text{bg}\}}} \sum_{i\in\mathbb{N}} \mathbf{1}_{\{c_i \neq \text{bg}\}} \cdot \ell_{huber}(x_i^{wh}, \hat{x_i^{wh}}) \tag{3}$$

Augmentation	Probability	Magnitude min	Magnitude max
Translation	0.5	0.05	0.15
Cropping	0.4	0.05	0.25

Table 1. **Data Augmentation** Probability and range of the magnitude for the application of translation and cropping augmentations.

Here, x_i represents the bounding box prediction for node v_i, $\hat{x_i}$ is the corresponding ground truth, and x^{wh} refers to the predicted width and height. The losses are computed only for the nodes with ground truth class $c_i \neq bg$, where bg refers to the background class.

Additionally, we compute the binary cross-entropy on the IoU confidence score to evaluate the confidence of each bounding box prediction. The confidence score is designed to be low if the IoU between the predicted bounding box and the ground truth is less than 0.5. The confidence loss is defined as,

$$l_{conf} = -\frac{1}{N}\sum_{i=1}^{N} [s_i \log(\hat{s}_i) + (1 - s_i)\log(1 - \hat{s}_i)] \tag{4}$$

where predicted confidence score $\hat{s}_i(x_i, \hat{x_i})$ is computed as,

$$\hat{s}_i(x_i, \hat{x_i}) = \begin{cases} 1 & \text{if IoU}(x_i, \hat{x_i}) \geq 0.5 \\ 0 & \text{otherwise} \end{cases} \tag{5}$$

Here, s_i represents the IoU confidence score of the bounding box prediction for node v_i.

1.1.3. Total loss

The total loss is a weighted summation of all individual losses,

$$l_{total} = \alpha l_{cls} + \beta l_{loc} + \gamma l_{dim} + \lambda l_{conf} \tag{6}$$

The loss weights are set as $\alpha = 1$, $\beta = 2$, $\gamma = 3$, and $\lambda = 1.5$.

1.2. Data Augmentation

We apply two types of data augmentations to train our model from scratch, with parameters summarized in Table 1. Since each training sequence must contain at least one bounding box, we randomly choose a bounding box as the anchor box and perform the augmentations around it.

SSL Block	mAP ↑	Params (M)↓
GCN	0.09	**4.7**
3D Isotropic SplineConv	**0.38**	34.9
2D Anisotropic SplineConv	0.36	5.6

Table 2. **Spatial Structure Learning:** 3D kernels provide a marginal performance improvement at a huge parameter expense.

The first augmentation performed is a random translation along the x and y coordinates, applied with a probability of 0.5. The maximum translation in each dimension is restricted between 5% and 15% of the input shape. The second augmentation is random cropping around the anchor bounding box with the probability of cropping set at 0.35. The cropping size is constrained to a minimum of 5% and a maximum of 25% of the input dimensions.

2. Network Architecture

The initial node features $\mathbf{x}$ are projected into a 16-dimensional space using an MLP$(4, 16)$. Each subsequent layer of the network processes the node features through an SMVL block, which combines features from the SSL and MVL components. At layer $\mathbf{n}$, the input features of shape $M_{in}^{\mathbf{n}}$ are transformed into a spatially and temporally aware node feature of shape $M_{out}^{\mathbf{n}}$. The network outputs have channels $M_{out} = (16, 16, 32, 32, 64, 64, 128, 128)$. The SSL block has a kernel size of $(8, 8, 1)$ and the MVL block has 4 heads in the backbone. For the detection head, the SSL block has a kernel size of $(5, 5, 1)$ with 1 head in the MVL block, which downsamples node features to a 64-dimensional shape. Batch normalization and ReLU activations are applied after each step but omitted in illustrations for conciseness.

3. Additional Experiments

While the main paper focused on experiments validating the importance of fusion and the impact of the detection head granularity in *eGSMV*, this section investigates each component of the SMVL block and the impact of other parameters from graph construction on the performance of our framework.

3.1. Ablation on Model Components

SSL Block. Here, we evaluate the impact of various spatial structure learning techniques for the SSL block. Presented in Table 2, the configurations compared are a standard GCN layer, an isotropic 3D spline kernel, and our proposed anisotropic 2D spline kernel. The MVL block is kept constant across all models for a fair comparison.

GCN serves as a baseline, aggregating spatial neighbors uniformly without spatial adaptiveness. The isotropic 3D kernel, similar to the approach used in previous works, introduces additional temporal depth, leading to an increased parameter count and computational overhead. Results indicate that both spline-based kernels outperform the GCN-based SSL. This could be attributed to GCN being more prone to overfitting than its Spline variants. Further, the 3D isotropic kernel adds a substantial number of parameters with only a marginal performance gain over the 2D variant. This suggests that while the spline kernel does well at capturing spatial features, the additional temporal context in the 3D setup provides limited benefit.

MVL Block	mAP ↑	Params (M) ↓
w/o motion vector features	0.33	5.6
w motion vector features (ours)	**0.36**	5.6

Table 3. **Motion Vector Learning:** Motion guidance improves performance in attention-based temporal learning.

Temporal aggregation	mAP ↑
Uniform aggregation	0.31
Single-head attention	0.34
Multi-head attention (ours)	**0.36**

Table 4. **Multihead attention:** Evaluating impact of different aggregation methods in the MVL block.

MVL Block. This ablation examines the impact of incorporating motion vector features within the MVL block. In Table 3, we compare two models: one with motion vector features encoded in the edge and one without. These results indicate that adding motion vector features improves mAP by 3% with a negligible increase in model size, demonstrating that motion vector guidance effectively enhances temporal representation without significant computational cost.

Next, we also evaluate the impact of multi-head attention with uniform aggregation as well as its single-head counterpart. As demonstrated through Table 4, multi-head attention (MHA) in MVL enables a different weighting to temporal neighbors that carry varying temporal motion cues due to differences in time and their spatial location. Temporal neighbors carry cues with lower inductive bias, unlike spatial neighbors.

3.2. Timing Experiments.

We compare the time our sparse and asynchronous method takes to process a new event against a dense method, which updates the entire graph on an Nvidia A30 GPU (Table 5). Since the SSL and MVL blocks operate on independent graphs, they can function in parallel, allowing improved efficiency. *eGSMV* needs $18.1ms$ on a graph with $4,000$ nodes and $20.9ms$ on a graph with $25,000$ nodes. With an increasing graph size, we observe a minimal increase in

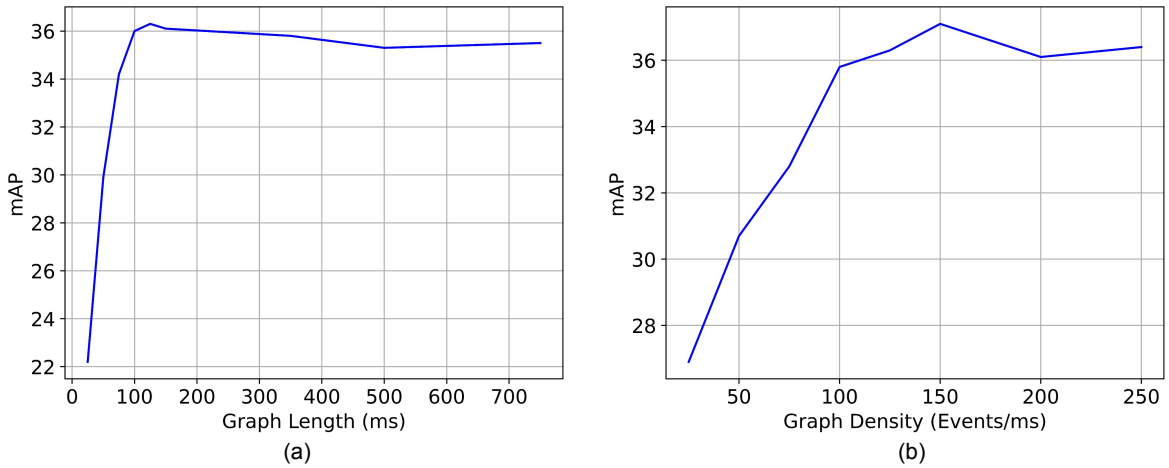

Figure 1. **Impact of Graph Construction on Performance.** (a) Variation in mAP with increasing graph length sequences, showing performance improvement up to an optimal sequence length before plateauing; (b) Variation in mAP with increasing graph density (number of events per ms), showing how performance improves with density up to a point before saturating.

Graph Update	2000	4000	10000	25000
Dense graph update	58.9	89.9	181.4	331.9
Serial (SSL × MVL)	24.4	25.1	27.4	29.9
Parallel (SSL × MVL)	**17.7**	**18.1**	**19.9**	**21.4**

Table 5. **Timing Experiments (ms) on increasing graph size:** Comparing time to process a new event by a dense graph update against our serial and parallel variants of asynchronous update.

time, unlike the dense graph update method. Additionally, our quadratic kernels enable more than **5x speedup** compared to AEGNN [15], which relies on cubic kernels. Finally, we believe that further caching optimizations, like in [6] and implementation on suitable hardware, could achieve greater improvements, as GPUs are primarily optimized for dense tensor operations.

3.3. Impact of Graph Length

In this experiment, we analyze the impact of graph length, defined as the time window, on the model's ability to capture spatial and temporal dependencies and, consequently, detection performance. A longer graph length accumulates more temporal context at the cost of increased computational requirements. However, a shorter graph with a very small look-back can miss critical temporal dynamics, particularly crucial in detecting slow-moving objects.

To evaluate this, we test the performance of our method on graph length sequences ranging from $25ms$ to $750ms$ as illustrated in Figure 1(a). We observe that graphs of a smaller sequence length underperform due to insufficient temporal context. The model achieves peak performance at a graph length of $100ms$, beyond which we observe diminishing returns in detection accuracy. This suggests that a look-back of $100ms$ strikes the optimal balance, capturing sufficient temporal information.

3.4. Impact of Graph Density

Graph density, determined by the number of nodes in a $1ms$ time window, directly influences the model's ability to aggregate meaningful features and manage computational requirements. This motivates our study of sampling density and its effect on performance. Aggressive sampling, which limits the number of nodes, may result in insufficient spatial and temporal interactions and reduce the model's ability to learn robust features. On the other hand, a denser graph provides more context per node, potentially enhancing spatial and temporal feature learning. However, excessive density increases computational complexity and memory requirements, potentially leading to overfitting.

We present a systematic variation in graph density by adjusting the number of permissible events per $1ms$ window from 25 to 250 as illustrated in Figure 1(b). We observe that moderately dense graphs achieve the best trade-off, enabling robust feature learning while having the least computational burden. This finding shows the importance of carefully tuning graph density to optimize performance in event-based vision tasks.

4. Additional Visualizations

We present qualitative results highlighting the detection performance across different datasets and scenarios, as shown in Figure 2. The visualizations provide insights into how *eGSMV* performs in different event distributions, motion dynamics, and background activity levels.

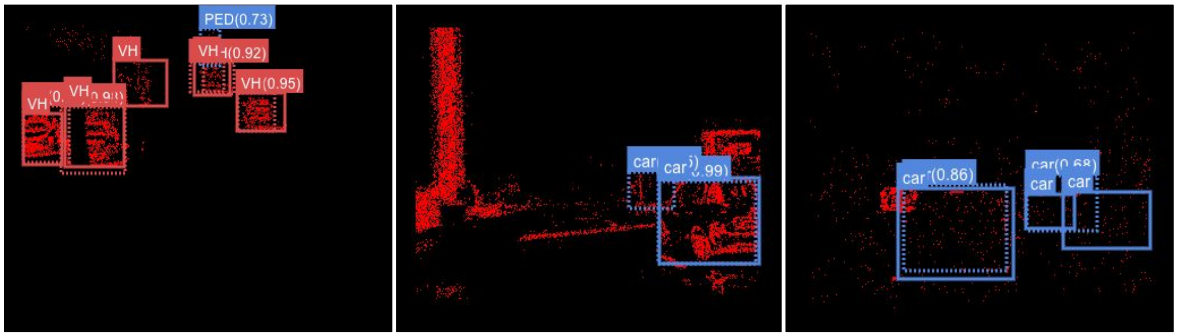

Figure 2. **Quantitative Results for Object Detection.** (a) Detection on the eTraM dataset, (b) Detection on the Gen1 dataset in an instance with egomotion, and (c) Detection on the Gen1 dataset in a stationary scenario and minimal events due to lack of motion.

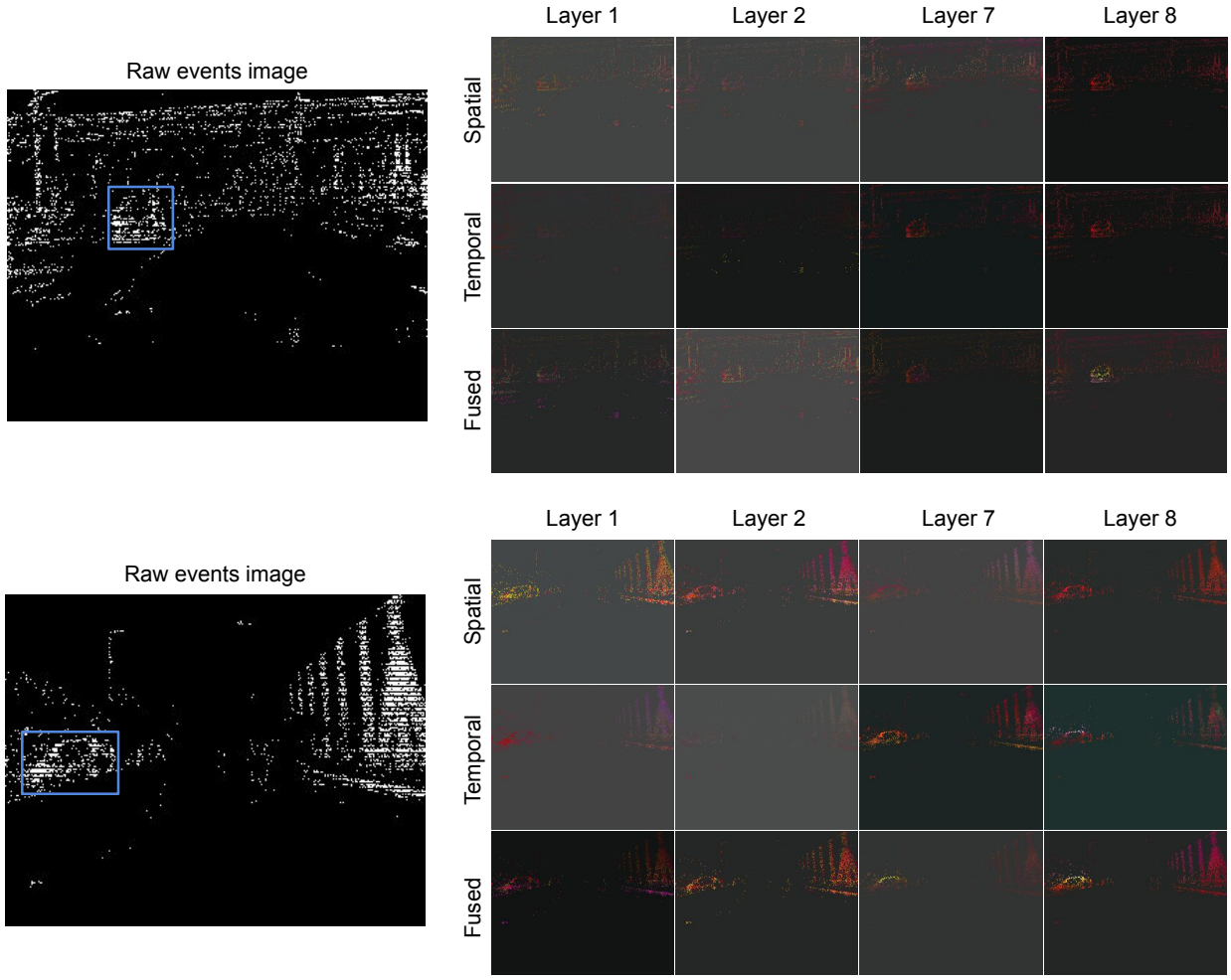

Figure 3. **Feature Maps during Inference.** Visualizations of the top three principal components (PCA) forming the RGB channels of the spatial (SSL), temporal (MVL), and fused feature maps. Feature maps chosen from the first two and final two SMVL layers.

Method	Type	Accuracy ↑
NVS-S [9]	Asynchronous GNN	0.67
AEGNN [15]	Asynchronous GNN	0.67
EvS-S [9]	Asynchronous GNN	0.76
AsyNet [12]	Asynchronous CNN	0.74
eGSMV	Asynchronous GNN	**0.78**

Table 6. **Object Classification on N-Caltech101**. Comparison of performance with relevant asynchronous methods.

Method	Type	Accuracy ↑
PointNet [14]	Sliding Window Voxels	0.88
AsyNet [12]	Asynchronous CNN	0.94
DVS-Gesture [1]	Synchronous CNN	**0.96**
eGSMV	Asynchronous GNN	0.94

Table 7. **Action recognition on DVS128-Gesture**. Comparison of performance with synchronous and asynchronous methods.

In Figure 2(a), detection results from the eTraM dataset demonstrate the localization capabilities in sequences with a sparse event distribution sequences and minimal background events. Figure 2(b) showcases the ability to capture spatiotemporal dependencies for accurate detection in high background activity cases when there is dynamic motion involved in the Gen1 dataset. Finally, Figure 2(c) presents detection results from the Gen1 dataset in a sparse event distribution under stationary conditions. Despite the challenges posed by limited event generation, the model, albeit with reduced confidence levels, successfully localizes the objects. These visualizations contain events from the most recent $25ms$.

Figure 3 provides a detailed visualization of how the architecture progressively localizes relevant objects in the scene across layers. The figure presents PCA-based feature maps for SSL, MVL, and fused representations from the initial and final SMVL layers. Initially, in the earlier layers, a majority of the events are assigned high importance, capturing broad spatial and temporal features. As the network progresses to deeper layers, the architecture refines its focus, selectively emphasizing events corresponding to the objects in the scene. As highlighted by these visualizations, this hierarchical learning process demonstrates the model's ability to integrate spatial and temporal dependencies effectively, allowing it to localize objects with increasing precision at deeper layers.

5. Generalization to Other Tasks

In this section, we assess the generalizability of the *eGSMV* backbone beyond object detection. We evaluate the method on two diverse event-based tasks: object classification (N-Caltech101 [13]) and action recognition (DVS128-Gesture [1]). Importantly, we use the same backbone with no architectural modifications.

5.1. Object Classification: N-Caltech101

The N-Caltech101 dataset evaluates static object classification from event streams. Table 6 shows that *eGSMV* achieves an accuracy of **78%**, outperforming previous asynchronous event-based baselines such as AsyNet, EVS-S, and AEGNN. This indicates that the proposed spatiotemporal multigraph backbone effectively transfers to the classification task.

5.2. Action Recognition: DVS128-Gesture

The DVS128-Gesture dataset evaluates dynamic hand gesture recognition which requires an understanding of the temporal change in the scene in addition to spatial understanding tasks like object classification. While most prior results rely on frame-based methods, we emphasize that *eGSMV* operates in an sparse and asynchronous manner. This avoids unnecessary computations and does not constrain the method for requiring the entire sequence to make an inference. Table 7 reports the accuracy of *eGSMV* and representative baselines. Our method achieves 94% accuracy, competitive with reported baselines, despite using no frame aggregation and no task-specific adjustments.

6. Spatiotemporal Graphs in Literature

Spatiotemporal graphs have also been studied in related contexts.Early formulations by [2] proposed modeling human activities in videos as spatiotemporal graphs, where nodes capture local features and edges encode their relations across time. More recently, [3] introduced spatiotemporal scene graphs for video question answering, linking objects and interactions across frames to enable higher-level reasoning. In a complementary direction, [10, 11] proposed the Graph of Events in Space and Time (GEST), which represents semantic events and their relations to connect vision and language in an explainable manner.

Our work differs in that the graphs are constructed directly from event-sensor data, with nodes corresponding to raw asynchronous events and edges capturing their local spatial and temporal relationships. While methods such as GEST target semantic abstraction and explainability in conventional video and language tasks, our focus is on efficient low-level vision representations that preserve fine temporal resolution and support real-time event-based downstream vision tasks. Despite these differences in requirements, these approaches highlight the versatility of spatiotemporal graphs, which can scale from fine-grained perceptual encoding to semantic reasoning.

7. Dataset Licenses

Gen1 [4] "Prophesee Gen1 Automotive Detection Dataset License Terms and Conditions": https://www.prophesee.ai/2020/01/24/prophesee-gen1-automotive-detection-dataset/

eTraM [16] "Creative Commons Attribution-ShareAlike 4.0 International License." https://github.com/eventbasedvision/eTraM

DSEC [7] "Creative Commons Attribution-ShareAlike 4.0 International License." https://dsec.ifi.uzh.ch

References

[1] Arnon Amir, Brian Taba, David Berg, Timothy Melano, Jeffrey McKinstry, Carmelo Di Nolfo, Tapan Nayak, Alexander Andreopoulos, Guillaume Garreau, Marcela Mendoza, Jeff Kusnitz, Michael Debole, Steve Esser, Tobi Delbruck, Myron Flickner, and Dharmendra Modha. A low power, fully event-based gesture recognition system. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7388–7397, 2017. 5

[2] William Brendel and Sinisa Todorovic. Learning spatiotemporal graphs of human activities. In *2011 International Conference on Computer Vision*, pages 778–785, 2011. 5

[3] Anoop Cherian, Chiori Hori, Tim K Marks, and Jonathan Le Roux. (2.5+ 1) d spatio-temporal scene graphs for video question answering. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 444–453, 2022. 5

[4] Pierre de Tournemire, Davide Nitti, Etienne Perot, Davide Migliore, and Amos Sironi. A large scale event-based detection dataset for automotive, 2020. 6

[5] Zheng Ge, Songtao Liu, Feng Wang, Zeming Li, and Jian Sun. Yolox: Exceeding yolo series in 2021. *arXiv preprint arXiv:2107.08430*, 2021. 1

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

[7] Mathias Gehrig, Willem Aarents, Daniel Gehrig, and Davide Scaramuzza. Dsec: A stereo event camera dataset for driving scenarios. *IEEE Robotics and Automation Letters*, 2021. 6

[8] Peter J Huber. Robust estimation of a location parameter. In *Breakthroughs in statistics: Methodology and distribution*, pages 492–518. Springer, 1992. 1

[9] Yijin Li, Han Zhou, Bangbang Yang, Ye Zhang, Zhaopeng Cui, Hujun Bao, and Guofeng Zhang. Graph-based asynchronous event processing for rapid object recognition. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 934–943, 2021. 5

[10] Mihai Masala and Marius Leordeanu. From vision to language through graph of events in space and time: An explainable self-supervised approach. *arXiv preprint arXiv:2507.04815*, 2025. 5

[11] Mihai Masala, Nicolae Cudlenco, Traian Rebedea, and Marius Leordeanu. Explaining vision and language through graphs of events in space and time. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2826–2831, 2023. 5

[12] Nico Messikommer, Daniel Gehrig, Antonio Loquercio, and Davide Scaramuzza. Event-based asynchronous sparse convolutional networks. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part VIII 16*, pages 415–431. Springer, 2020. 5

[13] Garrick Orchard, Ajinkya Jayawant, Gregory Cohen, and Nitish Thakor. Converting static image datasets to spiking neuromorphic datasets using saccades, 2015. 5

[14] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 652–660, 2017. 5

[15] Simon Schaefer, Daniel Gehrig, and Davide Scaramuzza. Aegnn: Asynchronous event-based graph neural networks. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2022. 3, 5

[16] Aayush Atul Verma, Bharatesh Chakravarthi, Arpitsinh Vaghela, Hua Wei, and Yezhou Yang. etram: Event-based traffic monitoring dataset. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 22637–22646, 2024. 6

[17] Zhaohui Zheng, Ping Wang, Dongwei Ren, Wei Liu, Rongguang Ye, Qinghua Hu, and Wangmeng Zuo. Enhancing geometric factors in model learning and inference for object detection and instance segmentation. *IEEE transactions on cybernetics*, 52(8):8574–8586, 2021. 1