# Neuromorphic Vision-based Motion Segmentation with Graph Transformer Neural Network

Yusra Alkendi<sup>1</sup> (D), Rana Azzam<sup>2,4</sup> (D), Sajid Javed<sup>2,5</sup> (D), Lakmal Seneviratne<sup>2</sup> (D), and Yahya Zweiri<sup>2,3,4</sup> (D)

Abstract—Moving object segmentation is critical to interpret scene dynamics for robotic navigation systems in challenging environments. Neuromorphic vision sensors are tailored for motion perception due to their asynchronous nature, high temporal resolution, and reduced power consumption. However, their unconventional output requires novel perception paradigms to leverage their spatially sparse and temporally dense nature. In this work, we propose a novel event-based motion segmentation algorithm using a Graph Transformer Neural Network, dubbed GTNN. Our proposed algorithm processes event streams as 3D graphs by a series of nonlinear transformations to unveil local and global spatiotemporal correlations between events. Based on these correlations, events belonging to moving objects are segmented from the background without prior knowledge of the dynamic scene geometry. The algorithm is trained on publicly available datasets including MOD, EV-IMO, and EV-IMO2 using the proposed training scheme to facilitate efficient training on extensive datasets. Moreover, we introduce the Dynamic Object Mask-aware Event Labeling (DOMEL) approach for generating approximate ground-truth labels for event-based motion segmentation datasets. We use DOMEL to label our own recorded Event dataset for Motion Segmentation (EMS-DOMEL), which we release to the public for further research and benchmarking. Rigorous experiments are conducted on several unseen publicly-available datasets where the results revealed that GTNN outperforms state-of-the-art methods in the presence of dynamic background variations, motion patterns, and multiple dynamic objects with varying sizes and velocities. GTNN achieves significant performance gains with an average increase of 9.4% and 4.5% in terms of motion segmentation accuracy (IoU%) and detection rate (DR%), respectively.

*Index Terms*—Neuromorphic Vision, Dynamic Vision Sensor, Event Camera, Motion Segmentation, Graph Transformer Neural Networks.

### I. INTRODUCTION

Scene understanding constitutes a cornerstone to a plethora of robotic applications where automation and behavioral intelligence are essential, such as navigation [1, 2], exploration [3], and simultaneous localization and mapping [4]. It incorporates perceiving sensory information to infer geometric and semantic properties of objects present in the context of a working environment. The robustness of such perception modules heavily depends on their ability to cope with various

<sup>1</sup>Yusra Alkendi is with the Propulsion and Space Research Center (PSRC) at the Technology Innovation Institute (TII), Abu Dhabi, UAE, e-mail: {Yusra.Alkendi@tii.ae}

<sup>2</sup>Rana Azzam, Sajid Javed, and Lakmal Seneviratne are with the Khalifa University Center for Autonomous Robotic Systems (KUCARS), Khalifa University of Science and Technology, Abu Dhabi, UAE

<sup>3</sup>Yahya Zweiri is also with the Advanced Research and Innovation Center (ARIC), Khalifa University of Science and Technology, Abu Dhabi, UAE.

<sup>4</sup>Rana Azzam, and Yahya Zweiri are also with the Department of Aerospace Engineering, Khalifa University of Science and Technology, Abu Dhabi, UAE.

<sup>5</sup>Sajid Javed is also affiliated with the Department of Computer Science, Khalifa University of Science and Technology, Abu Dhabi, UAE.



Fig. 1: Motion segmentation results of the proposed learning-based algorithm (GTNN) using EV-IMO (Floor sequence) publicly available dataset [15]. (A) APS image for visualization only. (B) Approximate Ground truth events: black represents foreground events and red represents background events. (C) Segmented events using the proposed GTNN algorithm: black indicates predicted foreground events and gray indicates predicted background events. Our GTNN performs a binary classification to differentiate between foreground events due to moving objects or background events due to camera motion.

inherent environmental challenges, such as scene dynamics, varying lighting conditions, and the absence of prior knowledge related to the number of objects. According to [5, 6, 7], approaches that can tackle such challenges are still under investigation. Scene Segmentation can be broadly divided into three main categories: Semantic [8], Instance [9, 10], and Motion Segmentation [11]. A recent progress in this area can be contributed towards advanced deep learning approaches exhaustively reviewed in [12]. Depending on specific application requirements, appropriate segmentation techniques can be selectively employed for accurate scene analysis. In a navigation scenario, where a robotic vehicle moves in its task environment and is anticipated to encounter one or more dynamic objects, motion segmentation is pivotal to ensure safety of the vehicle and its surrounding environment, and hence ascertain the continuity of the robotic task. Motion segmentation is defined as the segregation of some dynamic object, moving independently, from the background motion, based on observations acquired from passive [13] or active [14] sensors.

Vision-based motion segmentation is an active research area and several approaches have been proposed in the literature [16]. Neuromorphic vision is an emerging research area and technology that mimics the working principle of the human retina, by timely and asynchronously capturing the polarity of the log intensity variations in the observed scene at the pixel level. Neuromorphic vision sensors are also referred to as event-based cameras<sup>1</sup> acquire information in the form of continuous streams at low latency (>20 $\mu$ s), high temporal resolution (>800kHz), high dynamic range (>120 dB), and no motion blur [16]. Event-based cameras outperform the ca-

<sup>&</sup>lt;sup>1</sup>These two terms (Neuromorphic vision sensors and event-based cameras) will be used interchangeably throughout the paper.

pabilities of conventional cameras and therefore make them an ideal alternative for conventional vision sensors, particularly in applications that require robust perception, such as motion estimation of vehicles under varying illumination conditions in the presence of dynamic scenes. A wealth of event-based computer vision literature has unlocked new opportunities and initiated research directions for more robust visual perception. Such capabilities have been demonstrated in various applications such as visual localization [17, 18], depth estimation [19], object tracking [20], precision manufacturing [21], and tactile sensing [22], to name a few. Though event cameras have demonstrated significant advantages over their conventional counterparts, respective computer vision paradigms must handle the data sparsity and low spatial resolution typical of the acquired event streams.

Recently, few research work are conducted to develop motion segmentation paradigms based on data acquired from event cameras. These paradigms can be classified into: (1) classical approaches such as [23, 24, 25], and (2) learningbased approaches such as [26, 27]. In the former, motionmodel fitting and clustering methods are used to differentiate the events that belong to a moving object in the scene from those belonging to the background. In the latter, classification or regression using neural networks, such as spike and graph convolutional, are employed to identify events belonging to a moving object. These approaches either require prior knowledge of the scene (number of objects) and camera motion, camera parameter tuning, sensor data pre-processing (filtering, offline parameter computation, etc.), or initialization of the optimization problem. These limitations hinder the generality and applicability of the state-of-the-art (SOTA) event-based segmentation approaches and hence, this problem remains unsolved.

In the current work, we propose a learning-based motion segmentation algorithm<sup>2</sup> to aid robotic navigation in unknown dynamic environments. Particularly, we employ a Graph Transformer Neural Network (GTNN), based on the point transformer layer [28], to classify events triggered in a dynamic scene into moving-object events or background events. Our GTNN approach is based on processing raw event streams, acquired by an event camera, without requiring any prior knowledge about the topology or the dynamics in the scene, prior knowledge of the camera motion, camera parameter tuning, event pre-processing, or offline initialization. Eliminating such requirements is essential to achieve better generalizability across various scenes, motion scenarios, and sensors.

Transformers have been recently used for event-based applications and have demonstrated outstanding performance in processing event streams for event-denoising [29] object detection [30], and action and gesture recognition [31]. However, it is worth noting that the transformer operates on 2D frames in [31, 30]. These frames were reconstructed from events, as opposed to the work in [29] where a graph-driven transformer operates on raw event data. A point transformer has recently been proposed to handle dense representations of point clouds for object classification and segmentation using 3D data [28]. In the current work, we demonstrate the scalability of GTNN using the point transformer layer and address the challenges posed by unconventional and noisy sensor output, as well as the low-spatial resolution and sparsity of neuromorphic vision sensors. More specifically, working with event streams is deemed challenging due to the limited features encoded in raw event data, such as the spatial coordinate of the pixel where the intensity change occurred in the scene, the time of the event, and the polarity; +1 or -1 indicating an increase or decrease in intensity, respectively. To handle the differences in input data between the sparse representation of the event stream and the dense representation of a point cloud, a new architecture for the GTNN algorithm has been developed. This architecture includes a point transformer layer, as well as some additional changes in the number of encoder and decoder units and the addition of processing layers such as global aggregation. While the direct implementation of the point transformer structure developed by [28] was initially considered, evaluations have shown that it was too complex and less efficient for the event-based motion segmentation task at hand.

The proposed GTNN algorithm is trained in a supervised manner on publicly available datasets [15, 25, 32] to perform end-to-end motion segmentation, i.e GTNN takes as input unprocessed event streams and outputs the segmentation results. Evaluation sequences from the publicly available datasets are used to validate the proposed algorithm and to quantitatively and qualitatively compare it against the SOTA learning-based motion segmentation approach [26] and the offline classical approaches [24, 25, 33]. The segmentation accuracy achieved by GTNN outperforms SOTA motion segmentation approaches [26, 24, 25, 33]. Further to that, the GTNN model is more efficient than [26] in terms of computational requirements and hence performs faster prediction. The testing scenarios exhibited various scene dynamics, changing illumination conditions, and different camera motion dynamics. The proposed algorithm is robust against all of the aforementioned challenges and requires no fine-tuning upon testing in unknown environments using unseen sequences. Fig. 1 shows sample segmentation results obtained when testing our proposed algorithm on a publicly available dataset of EV-IMO (Floor) [15].

Additional testing sequences are recorded in our lab facilities to further analyze the performance of the proposed algorithm in scenarios recorded in a different domain than that of the publicly available training and testing datasets. To that end, we propose the Dynamic Object Mask-aware Event Labeling (DOMEL) approach to generate approximate groundtruth event labels for the recorded sequences. The performance of GTNN on these recorded sequences has verified its generalization across new domains.

The overall framework proposed in this paper is depicted in Fig. 2 highlighting the detailed design of (1) the graph transformer neural network (GTNN) for motion segmentation and (2) the dynamic object mask-aware event labeling (DOMEL).

To summarize, the contributions of this work are:

<sup>2</sup>A supplementary video is available at: <https://youtu.be/rVvbKdXh6oE>

1) The design and development of an event-based motion

segmentation algorithm, based on GTNN. The proposed GTNN (1) preserves the asynchronous nature of event streams and exploits spatiotemporal correlations to infer the scene and camera motion dynamics, (2) does not require any prior knowledge about the scene geometry and/or dynamics, and (3) does not need initialization or event pre-processing to perform motion segmentation.

- 2) The design of an effective training scheme that facilitates training on an extensive dataset while reducing computational requirements. This scheme enables faster convergence and hence higher performance than conventional training by exposing portions of the training data to the network at every iteration.
- 3) Extensive evaluation of the proposed GTNN algorithm on publicly available event datasets and on experimental sequences recorded locally in our lab facilities. Comparisons to the SOTA learning-based and classical motion segmentation approaches have also shown the superiority of the proposed algorithm in terms of segmentation accuracy (IoU%) and detection rate (DR%).
- 4) The release of a new event dataset (EMS-DOMEL) with the corresponding motion segmentation ground truth labels obtained using the proposed DOMEL approach. Data is recorded using two event cameras with different resolutions, in scenes involving multiple dynamic objects of various sizes/types in a variety of challenging environmental scenarios.

The dataset can be accessed through the following link: <a href="https://github.com/Yusra-alkendi/EMS-GTNN">https://github.com/Yusra-alkendi/EMS-GTNN</a>> for further research and benchmarking.

The remainder of this paper is organized as follows. Section II provides a review of the related literature. Section III describes in detail the proposed GTNN algorithm and the EMS-DOMEL dataset. In Section IV, the experimental results are presented, analyzed, and compared to SOTA motion segmentation approaches. The conclusions of the current work are drawn in Section V.

# II. BACKGROUND AND RELATED WORKS

## A. Event-based Motion Segmentation

Solutions for event-based motion segmentation have been investigated throughout the past years for variable environmental conditions at different complexity levels. Owing to the camera's working principle, events are only generated if changes occur, either due to camera motion or dynamics in the scene. For the simplest scenario where the event camera is static, motion segmentation has been tackled using clustering approaches as in [34, 35, 36].

In case the camera capturing the scene is moving, events are generated due to two main reasons: (1) camera egomotion, and (2) moving objects. To differentiate between these two categories, camera motion should be estimated and/or additional information about the environment is required [37]. A recent motion segmentation framework based on motion compensation has been proposed in [23], where raw events are accumulated into 2D frames. The frames are then aligned with high contrast contours, and evaluated using dispersion [33] or sharpness [38, 39] measures. For high dynamic environments, Mitrokhin et al. [33] have proposed an approach to detect moving objects by fitting a motion model using multi-key parameters into the background, and then considering misaligned events to belong to different segments/classes. The clustered objects are then tracked by a Kalman filter to handle occlusions and scene uncertainties. This approach might not work if multiple moving objects appear in one scene. This work was later extended in [23] by integrating the Expectation-Maximization (EM) approach in the segmentation method. Their results showed excellent segmentation and optical flow estimation. However, the EM-based segmentation algorithm requires prior knowledge about the scene, particularly the number of moving objects to start initialization.

Recently, Parameshwara et al. [25], have proposed an approach using a combined nonlinear optimization method to segment multiple objects moving independently, without knowledge of the number of moving objects in the scene. Furthermore, Mitrokhin et al. [15] proposed a learning-based framework for motion segmentation using motion compensation, where depth, ego-motion, and clustering of independent moving objects (IMOs) and their 3D velocity were estimated.

A recent study by Zhou et al. [24] developed an offline optimization approach based on energy minimization where identification of IMOs acquired with an event-based camera is performed based on motion fitting and clustering methods. The iterative scheme allows for exploiting spatiotemporal correlations between event streams for event identification. The proposed algorithm requires no prior information about the scene, dynamic objects within the scene, or camera motion. It, however, requires an initialization stage to start the optimization.

Lastly, Parameshwara et al. [26] developed a learning-based motion segmentation approach using an encoder-decoder spike neural network architecture, called a SpikeMS model. SpikeMS is a binary classifier based on a novel spatiotemporal loss function. SpikeMS is capable of performing incremental predictions for event streams where the segmentation accuracy is comparable to offline classical-based approaches [33, 23, 25]. Spatiotemporal correlation methods and deep learning approaches have shown potential in the reviewed event-based motion segmentation approaches, however, various aspects of this field remain largely unexplored.

## B. Graph Transformers Neural Networks

Deep learning models which operate on non-Euclidean graphstructured data are known as graph neural networks (GNNs). GNNs have been successfully employed in a multitude of applications [40] such as object segmentation [41] due to their expressive power and modeling flexibility. GNNs perform mapping of input graphs based on their node features and connectivity within a neighborhood (edges), despite the order in which they are fed to the neural network. It is also worth noting a single GNN architecture may accept input graphs of variable sizes. This important property of GNNs highly suits the asynchronous nature and variable input rate of event streams, which are a function of the scene and camera motion dynamics.

Transformers, on the other hand, have recently proven



Fig. 2: Proposed event-based motion segmentation framework based on graph transformer neural network (GTNN). GTNN is developed and trained using the proposed training scheme on publicly available event datasets (EV-IMO, MOD, EV-IMO2) and tested on the corresponding evaluation sequences along with our recorded experiments (EMS-DOMEL). The approximate ground truth event labels of our recorded experiments are generated using the proposed DOMEL approach. The proposed algorithm classifies incoming event streams into foreground events related to moving object(s) or background events.

cutting-edge performance in a variety of applications, including natural language processing [42] and computer vision tasks [43, 44, 45, 46, 47]. The self-attention head used in transformers is responsible for capturing the relationships between inputs and outputs and allowing simultaneous processing of sequential recurrent networks. Graph-based transformers with their self-attention operation have proven an outstanding capability for processing 3D data such as 3D point cloud [48, 28, 49, 50] for vision tasks like segmentation and classifications. Inspired by this success in 3D data processing and our previous work on GNN-transformer for events denoising [29], we design a learning-based motion segmentation network based on a graph transformer neural network. The proposed algorithm allows for handling the asynchronous nature of events for revealing their spatiotemporal correlations and processing the scene dynamics accordingly. An event stream is fed to the algorithm, which extracts the local features of every event and integrates them with a global feature representing the whole stream, prior to processing them using nonlinear operations. The event stream is then segmented into events that belong to the background and others that belong to moving objects in

the scene.

#### **III. PROPOSED FRAMEWORK**

In this section, the proposed design of (1) the graph transformer neural network (GTNN) for motion segmentation and (2) the dynamic object mask-aware event labeling (DOMEL) are presented in detail.

# A. Graph Transformer Neural Network (GTNN) Algorithm for Event-based Motion Segmentation

The GTNN motion segmentation algorithm operates on incoming event streams, acquired through a moving event camera in a dynamic scene to carry out classification of events into (1) foreground events: events that belong to dynamic objects in the scene, and (2) background events: events that belong to the static background and were generated due to camera ego-motion. GTNN processes events in their raw formats, i.e. does not perform any event pre-processing such as accumulation into 2D frames, and hence preserves their asynchronous nature. Particularly, events are structured as 3D graphs encapsulating their spatiotemporal properties. Event graphs are then passed through a set of convolution and deconvolution operations as per the architecture visualized in Fig. 3.

The remainder of this section presents in great detail the implementation of the proposed GTNN algorithm. Section III-A1 provides details on events-3D graph construction based on k-nearest neighborhood (kNN) strategy, Section III-A2 explains the events-3D graph transformation process where the graph nodes and their features are nonlinearly mapped by the encoding-decoding GTNN model, and Section III-A3 delineates the architecture of the proposed GTNN.

### 1) Events-3D graph construction

Data perceived by an event camera report changes in log intensities of the observed scene in the form of asynchronous events. Events span a spatial resolution of  $H \times W$  pixels, where H and W are the vertical and horizontal dimensions of the camera's frame. A stream of N events,  $\{e_i\}_N$ , can be represented using a sequence of 4-tuples as indicated below:

$$\{e_i\}_N = \{x_i, y_i, t_i, p_i\}_N,\tag{1}$$

where  $(x_i, y_i)$  is the event's pixel coordinate at which event *i* has occurred within the frame,  $t_i$  refers to the time at which event *i* is triggered, and  $p_i$  is the event's polarity. Event polarity may take one of two values: +1 if the brightness of the pixel has increased and -1 otherwise.

A stream of events triggered within a pre-defined temporal window is structured as a 3D graph G. In case of aggressive vehicle maneuvers, and due to the high temporal resolution of the event camera, a huge amount of event pixels will be active simultaneously. Consequently, inferring spatiotemporal event relationships is deemed challenging due to data redundancy and high computational requirements. To resolve this issue and to minimize memory usage, the size of the graph constructed from the temporal window is limited to a fixed maximum  $(N_{max}=5000)$ , which was decided based on visual assessments of the events' projection on a 2D frame. This means that the temporal window could be shorter if too many events were received as a result of high scene dynamics, in which case the most recent  $N_{max}$  events will be used to construct the graph. As for memory requirements, classification of an event sample requires storing up to  $N_{max}$  events. It is worth noting that graphs constructed from a single event sequence may exhibit variable sizes at different times, depending on the scene and camera dynamics. Operation on variable-sized graphs is a property of graph-based neural networks. By virtue of this, our proposed approach may generalize across various domains, regardless of the number of triggered events in the temporal window.

Each node in the graph G represents an event in the stream. Consequently, node features are the properties of the triggered events, namely  $\langle (x_j), (y_j), (t_j) \rangle$ , where j represents the node index in the graph,  $(x_j, y_j)$  are the event's pixel coordinates, and  $t_j$  is the event's timestamp. Although event polarities may provide insights about the motion direction through analysis of brightness changes, this property is sensitive to the camera's parameter settings. To that end, performing motion segmentation based on this event property may limit the generality of the proposed algorithm, hence why we omitted it from the graph node features. A similar consideration was adopted in our previous work [29].

Next, the k-Nearest Neighbor (kNN) search is performed to connect every node  $(e_i)$  to k-nearest neighboring nodes  $(e_j)$  based on their 3D spatiotemporal distances. Resulting spatiotemporal neighborhoods are referred to as sub-graphs, each containing k + 1 nodes. These sub-graphs will be further processed by the encoding-decoding operations as will be discussed in Section III-A2.

2) Events-3D graph transformations

The constructed 3D graph, including all the sub-graphs, will be passed through a set of encoding and decoding layers in GTNN. In this section, the details of the core layers will be explained; namely a point transformer layer, transition down module, and transition up module.

**Point Transformer Layer:** The point transformer layer operates on the sub-graphs generated in the previous step. It is worth mentioning that several implementations of transformer layers for classification and segmentation of 3D data were found in the literature, such as [28, 48]. The implementation adopted in our proposed approach was inspired by [28], since the constructed 3D event graphs exhibit the same structure as 3D point clouds, although the data inference is different.

The point transformer layer implements a self-attention mechanism that captures the sequence relationship between the inputs and outputs for structured prediction tasks. The attention mechanism based on the query–key–value (QKV) model enables functions with high long-term memory [42] and executes parallel processing which can reveal jointly complex relationships between inputs and outputs. The most popular self-attention operators are scalar attention [42] and vector attention [51]. The vector or dot-product attention operator (i.e. a simple matrix multiplication) is selected in our algorithm to achieve faster state update and better space efficiency. The point transformer layer is composed of a residual block to perform feature aggregation and feature transformation as illustrated in Fig. 3-(1).

Let X(i) be a set of input feature vectors. X(i) will be processed by three connected operation streams. In the first stream operation, a subtraction relation is used between the input features after being transformed in a point-wise manner by  $\varphi$  and  $\psi$ . Those will be added with a position encoding  $\delta$ (from the second operation) and consequently forming the attention vector which is nonlinearly mapped by  $\gamma$  through MLP. Concurrently, in the third operation stream, input features are transformed by  $\alpha$  and added to a position encoding  $\delta$ . Then, the outputs of all stream operation lines are aggregated using the Hadamard product.  $\varphi$ ,  $\psi$ , and  $\alpha$  are point-wise feature transformations, such as linear projections or nonlinear MLPs.  $\delta$  is a position encoding function.  $\gamma$  is a mapping function (in our case, two layers of MLP, followed by ReLU). The output of the point transformer layer is represented by the following formula:

$$y_i = \sum_{x_j \subseteq X(i)} \rho(\gamma(\varphi(x_i) - \psi(x_j) + \delta)) \odot (\alpha(x_j) + \delta), \quad (2)$$

where  $y_i$  is the output feature.  $\rho$  is a normalization function. Spatiotemporal patterns between a set of events are essential



Fig. 3: Proposed Graph Transformer Neural Network (GTNN) Architecture for Event-based Motion Segmentation. The classification network takes N number of events as input stream, applies input and feature transformations and mapping, and then aggregates global features by global max pooling. The output is a binary classification assigned to each event in the stream, indicating whether it belongs to class 0 (a moving background due to camera motion) or class 1 (dynamic object(s) within the scene).

to carry out event-based motion segmentation. Such patterns could be obtained based on extracted local and global feature correlations between events in a graph. By operating on subgraphs, the point transformer layer will extract information on the local coherence between the events. This local graph operated-attention transformer has been adopted in literature for image analysis [51] and 3D point cloud processing [28].

The 3D event graph G is passed to the point transformer layer, where every sub-graph is processed to nonlinearly map the node features. Sub-graphs are referred to as X(i), where  $X(i) \subseteq X$ , and X is the 3D graph G in our implementation. The output of the layer is of the same structure as the input graph G. Details on the point transformer layer are depicted in Fig. 3-(1).

**Transition Down Module:** In this module, the cardinality of the 3D graph is reduced by a certain factor, to convolve the graph nodes. For instance, for a graph G with N nodes and a reduction factor 4 is requested, the transition down module will output a new graph containing N/4 nodes. As schematically illustrated in Fig. 3-(2), G(p1) refers to the input graph with p1 nodes that enters the transition down module where G(p2) is the output graph (reduced graph size) with p2 nodes. The farthest point sampling (FPS) algorithm is adopted and performed in G(p1) to identify a well-spread downsample subset G(p2) (where  $G(p2) \subset G(p1)$ ) with the requested cardinality. It is worth noting that the pooling is performed on G(p1) using the same kNN graph strategy to obtain G(p2), which is the same neighboring set of data previously identified in the point transformer layer. In this work, k is selected to be 16 based on the ablation study which is performed on this hyperparameter and its variants. In this module, each input feature is passed sequentially through a linear transformation, batch normalization, and ReLU operations. Then a max pooling is operated onto each node in G(p2) based on k neighborhood nodes in G(p1).

Transition Up Module: U-net architecture is adopted [28] where the encoder layers, in our case point transformer layer, are coupled with the corresponding decoder layers. This coupling was adopted since motion segmentation is similar to semantic segmentation which requires a dense prediction-masked output. The main function of the transition up module is to map features from the reduced graph data set, G'(p2), onto its superset graph data set, G'(p1) (where  $G'(p1) \supset G'(p2)$ ). Similar to transition down operation, each input point feature is sequentially processed by a linear layer, batch normalization, and ReLU operations. The node features of G'(p2) are mapped to a high-dimension graph size, G'(p1), using trilinear interpolation. To that end, the interpolated features of G'(p1) and the features of corresponding encoder stage G'(p1) are concatenated via the skip connection. In other words, and as mentioned earlier, the point transformer layers operate as the network encoder layers where their output graph is connected to the output graph from transition up modules (decoders) via skip connection. The structure of the transition up module is depicted in Fig. 3-(3).



Fig. 4: Encoding operation of a sample 3-D event graph by GTNN encoder unit, a coupled point transformer layer (stage 1) and a transition down module (stage 2).

#### 3) The GTNN architecture

Fig. 3 shows the overall architecture of GTNN, where given an input 3D graph, node features will be processed by means of various nonlinear operations to perform motion segmentation. In other words, every event (i.e. node) in the graph will be classified as a foreground event or a background event.

The event graph is encoded in two stages, as depicted in Fig. 4; (1) node features are encoded using the point transformer layers and (2) graph nodes are encoded using the transition down module. A point transformer layer coupled with a transition down module is referred to as an encoder unit. Similarly, a point transformer layer coupled with a transition up module is referred to as a decoder unit.

Based on the application and dynamics in the data, the number of encoder units is varied. In our case and according to the ablation study, the best-performing network architecture has three encoder-decoder units. The selected downsampling rates for the transition down modules are [1, 4, 4] which correspond to graph sizes of [N, N/4, N/16], where N is the number of events within the graph G. Consequently, the upsampling rates for the corresponding transition up modules will be [4, 4, 1]. Hence, the output graph will be of the same size as the input graph.

Obtaining a global feature vector that correlates the events in the graph is necessary for motion segmentation. This is achieved by transitioning down the graph nodes, obtained after the three encoder layers, into a single node. The features of this node are passed through an MLP and transformed by a graph average pooling operation forming a global feature vector. Inspired by the other segmentation models [48], the global feature vector is appended to every node feature in the graph after the last decoder layer in the network. The output is then passed to an MLP followed by a softmax (Eq. (3)) layer that generates a  $N \times 2$  tensor. The entries in each row are the predicted probabilities of an event being a foreground or a background event.

$$\mathbf{Softmax}(x_i) = \frac{e^{x_i}}{\sum_{j=1}^2 e^{x_j}} \tag{3}$$

A detailed ablation study is presented in the supplementary material where the number of encoder-decoder units was varied, and the performance of the network with and without aggregation of the global feature vector was analyzed.

## B. Proposed Effective Training Scheme

In our work, a neural classifier is trained on set of 3D graphs, representative of event streams to perform motion segmentation in a supervised manner. The network will carry out node classification, where it will predict a label/class for every node in the graph. Given an input 3D graph and the true node labels, the weights of the classification neural network are optimized to minimize the difference between the predicted output and the ground truth label. The sparsity and the limited information obtained from event data, i.e.  $\langle x_i, y_i, t_i, p_i \rangle$ , make scene interpretation and hence, end-to-end vision tasks such as motion segmentation challenging. On the plus side, the high temporal resolution of event data makes it possible to obtain large amounts of data in a short period of time.

The accuracy of deep learning models heavily depends on the quality and quantity of the datasets used for training and evaluation. Exposing neural networks to large amounts of diverse training data is essential for (1) effectively tuning the network's parameters to generate accurate predictions, and (2) enhancing the network's ability to generalize well to unseen samples. Nevertheless, extensive efforts are required to obtain, for some applications, and correctly annotate large amounts of training data. Moreover, operating on large training datasets is computationally expensive and models may not seamlessly converge to the global optimum solution. To circumvent these challenges, researchers have investigated various techniques [52, 53] to optimize the selection of training data to improve training efficiency.

In this work, we propose an effective training scheme, to exploit the availability of huge amounts of training data to facilitate global convergence, while reducing the training computational requirements, as shown in Fig. 5 and Algorithms 1. Instead of using the full training dataset every epoch, we propose to split it into L subsets. In every iteration, the network is exposed to only one of the training subsets, while other subsets remain idle. Consequently, the network will train, i.e. carry out backpropagation, on a particular subset once every L iteration. Therefore, the neural network will be exposed to the full training dataset after L training epochs. This approach differs from conventional training methods, in which the entire training dataset is typically used to update the weights of the neural network in every epoch using the minibatch gradient descent method. Specifically, in this method, a subset of the dataset (called a mini-batch) is exposed to the network, the error is calculated, and the weights are updated accordingly. However, due to the large size of the graph nodes and the limited memory capacity of the computer, the mini-batch size in our case is limited to 8. In both the conventional and proposed effective schemes, we use the minibatch gradient descent method.

However, the difference between the conventional approach (Algorithm 2) and our proposed method (Algorithm 1) is that the mini-batch gradient descent is implemented on a subset, rather than the entire dataset. This allows us to take advantage of the benefits of mini-batches when training a network on large data samples.



Fig. 5: The framework of the proposed effective training scheme compared to the conventional training scheme. Note that *i* is the current epoch number, *L* is the number of training subsets, J = i%L where % is the modulo operator.

Algorithm 1 Effective Training Scheme with Data-Dropout Strategy

- 1: Let L be the total number of subsets
- 2: Let N be the total number of epochs
- 3: for i = 0 to N 1 do
- 4: Compute subset index  $J = i \mod L$
- 5: Extract subset  $S_J$  from the training set
- 6: Train the model using subset  $S_J$
- 7: end for

Algorithm 2 Conventional Training Scheme without Data-Dropout Strateg

- 1: Let N be the total number of epochs
- 2: for i = 0 to N 1 do
- 3: Train the model using the entire training dataset
- 4: end for

To show the effectiveness of the proposed training scheme, it was used to train GTNN and then compared to the conventional training scheme on the same training dataset, namely MOD [25] dataset. MOD contains sequences of event streams, for benchmarking learning-based motion segmentation models. Fig. 6 depicts the loss curves obtained in both cases where it can be clearly observed that the proposed effective training scheme has resulted in faster training and better network learning capacity. Although the loss value obtained using conventional training in the first 400 training epochs was less than that obtained using our proposed training scheme, the training time was five times slower as indicated in Table I. After 400 epochs, the loss was still decreasing using the proposed training scheme, which proves the enhanced network learning capacity. Upon testing the trained models on unseen data, the results demonstrated the effectiveness of the proposed training scheme where better segmentation accuracy was achieved compared to conventional training, as listed in Table I.



Fig. 6: Loss curves were obtained while training the network with the proposed effective and conventional training schemes (with and without data dropout strategy, respectively).

TABLE I: Comparison between the training execution time and testing performance using the proposed effective and conventional training schemes.

	Execution/Evaluation Metrics	Effective Training Scheme (with Data-Dropout strategy)	Conventional Training Scheme (without Data-Dropout strategy)						
	Training								
Large	Required time to execute one epoch (s)	42	225						
	Testing								
	(TP, FP, FN, TN)	(97681, 96816, 58823, 2216680)	(82264, 145343, 74248, 2168062)						
	F1 score	55.6%	42.8%						
	Recall score	62.4%	52.5%						

## C. Dynamic Object Mask-aware Event Labeling (DOMEL)

DOMEL is a new framework capable of generating annotations for event streams from a variety of event camera types, including but not limited to DAVIS346c and DVXplorer. This flexibility is made possible by leveraging reference frames from the DAVIS346c camera, which provides both event streams and RGB images, unlike the DVXplorer that only captures event streams. By accommodating different sensing modalities, DOMEL ensures that event labeling can be consistently applied across diverse camera technologies, facilitating the development and testing of learning-based models for the task at hand.

To validate the generalization capability of the proposed GTNN motion segmentation algorithm, new experimental sequences are recorded in various domains using event cameras with different resolutions. For a thorough quantitative and qualitative evaluation of GTNN using these sequences, a labeling method must be devised to facilitate comparison of the GTNN prediction against the ground truth.

To that end, inspired by the success of the KoGTL [29], we propose Dynamic Object Mask-aware Event Labeling (DOMEL), which is an offline approach for annotating event data for motion segmentation applications. Every event in the recorded stream is assigned a label; foreground event or background event. The labeling process requires as input the corresponding gray-scale frame; which can be captured using a frame-based sensor, working simultaneously alongside the event camera. Hence, DOMEL allows labeling event streams recorded using event cameras that do not generate grayscale images such as DVXplorer. The frame-based sensor and the event camera need to visualize the same scene, with the same field of view.

In the following sections, the experimental setup for record-

ing the event dataset will be described and the labeling process will be explained in detail. Please refer to Fig. 2 for an illustration of DOMEL and how it fits in the overall proposed framework.

#### 1) Experimental setup

Two dynamic active pixel vision sensors; DAVIS346C and DVXplorer, are mounted side-by-side on a tripod, to capture a dynamic scene. DAVIS346C and DVXplorer have a spatial resolution of  $346 \times 260$  and  $640 \times 480$ , a bandwidth of 12 and 165 MEvent/s, and a dynamic range of 120 and 90 dB, respectively. The cameras are moved along various trajectories; i.e. a sequence of translations and rotations, in environments with various scene geometries where objects of various types and sizes are randomly moving.

Three measurements were recorded; (1) DAVIS346C event streams, (2) DVXplorer event streams, and (3) Grayscale images which capture intensity measurements of the dynamic scene. The grayscale images are obtained from the frame output of the DAVIS346C sensor and are denoted as active pixel sensor (APS) images hereafter. It is worth noting that in absence of the frame output of the event camera, it is possible to use a standard camera to capture the same scene alongside the event camera.

### 2) Labeling Framework

DOMEL approach includes four main stages, event-image synchronization, raw event-edge fitting, spatially shifted event-mask fitting, and event labeling, as illustrated in Fig. 7.

### a) Event-Image Synchronization

Synchronization of the recorded event streams from DAVIS346c and DVXplorer with the corresponding APS images is vital to the success of the proposed labeling technique, since events are matched to the corresponding APS frames captured at the same time. Synchronization is achieved by recording sensor data through a single ROS[54] node, and hence a common clock is used to record the measurements' timestamps, which can be seamlessly matched to obtain synchronized measurement pairs, as shown in Fig. 7-(I).

### b) Raw Event-Edge Fitting

The iterative closest point (ICP) fitting technique [55] is used to fit event streams to their corresponding APS canny [56] edge data. Due to the high temporal resolution of event data acquisition, fitting of events to edges is performed in several iterations. If the scene exhibits high dynamicity or the motion of the camera is fast, event streams are generated at a higher resolution than APS frames. Subsequently, events, particularly those in between two APS frames, would slightly deviate from the APS edges. To compensate for this deviation, ICP is used to perform a spatial shift (i.e., rotation and translation) to the events, which will facilitate matching them to APS edges, as presented in Fig. 7-(II). The spatially shifted events will be used for further processing in the next stages.

## c) Spatially Shifted Events-Mask Fitting

APS Frames capturing dynamic objects in the scene are first processed to generate masks of the objects. These masks could be obtained using any object masking algorithm, such as image segmenter [57]. The spatially shifted raw events are fitted to the corresponding masked-object frames using ICP, similar to stage (II), in several iterations for both cameras, as shown in Fig. 7-III. This is due to the high temporal resolution of the sensors, especially when capturing a dynamic scene, and the camera is in motion.

## *d*) *Event-Labeling*

In the last stage, events that are fitted to the corresponding masked-object frame are labeled as foreground events representing the moving objects captured when the camera is in motion (Class 1), as shown in Fig. 7-(IV). Whereas events that are not part of the masked pixels of the frame are considered background events (Class 0).

Our framework is efficient due to the use of ICP fitting which accommodates the high temporal resolution of DVS data acquisition. Events could slightly misalign with the edges in images because of slight timing mismatches between the time when events and image frames are captured. Therefore, ICP is used to precisely align the edges of event data and image pixels, correcting any slight shifts in their pixel positions. This ensures that our annotations accurately overlay the dynamic object mask, which is a significant improvement compared to other annotation schemes ([15, 32] that might not fix these small but important differences.

## 3) EMS-DOMEL Dataset

In this section, the collection and annotation of the Eventbased Motion Segmentation dataset (EMS-DOMEL) using the DOMEL framework is discussed. The dataset captures multiple independently moving objects in an indoor environment using a moving event camera. The sequences capture a variety of scenes, multiple objects moving at various speeds and in random paths, and unknown camera motions with various sensor resolutions. Table II presents a summary of event sequences captured using two cameras with different resolutions, detailing the number of detected objects, dynamic object events, and background events for each sequence. For instance, in DOMEL-Seq01 captured using a DAVIS346c camera, two moving objects were detected, resulting in 42,980 dynamic object events and 440,650 background events. Similarly, DOMEL-Seq07 to DOMEL-Seq09 which were captured using the DVXplorer camera, showcase varying numbers of moving objects resulting in different amounts of dynamic and background events as listed in the table. The ground truth labels facilitate rigorous assessment through metrics such as F1 score, Recall, Intersection over Union (IoU), and Detection Rate, which serve as evaluation benchmarks detailed in the following section (Section IV-B). This dataset is essential for advancing learning-based motion segmentation models, offering a wealth of data for both training algorithms and their corresponding performance evaluation metrics.

TABLE II: Summary of EMS-DOMEL Dataset.

EMS-DOMEL	Sensor Type	# Dynamic Objects	# Dynamic Object Events	# Background Events
Seq01	DAVIS346c	2	42,980	440,650
Seq02	DAVIS346c	2	52,176	494,378
Seq03	DAVIS346c	4	282,397	640,531
Seq04	DAVIS346c	2	37,220	146,949
Seq05	DAVIS346c	2	112,980	816,152
Seq06	DAVIS346c	2	66,989	560,096
Seq07	DVXplorer	3	616,467	2,923,756
Seq08	DVXplorer	3	889,215	5,953,718
Seq09	DVXplorer	1	77,314	4,435,217



Fig. 7: DOMEL framework for Event-based Motion Segmentation (EMS). DOMEL is a novel event labeling methodology developed to classify events, acquired when the camera is in motion, into two main classes: foreground or background events. The proposed DOMEL works irrespective of the sensor resolution, and hence any event camera may be used to record the event streams. A frame-based sensor is needed to capture intensity images corresponding to the recorded events, which will assist in event labeling.

# IV. EXPERIMENTAL EVALUATIONS A. Training and Testing Datasets

The proposed GTNN will be trained and evaluated using publicly available datasets including MOD, EV-IMO, and EV-IMO2 datatsets. MOD [25] is a simulated dataset targeted for learning-based motion segmentation models for event cameras. The environment captured in this dataset is a highly textured synthetic indoor room, where one to three dynamic objects appear intermittently, while the camera is moving. EV-IMO [15] dataset, on the other hand, contains various event sequences recorded in a real lab environment. The sequences exhibit varying levels of complexity in terms of the motion of the camera, dynamic objects' number, motion and speed randomness, and occlusion, feature-rich background texture, and a range of lighting conditions in the lab. Five different sequences of the EV-IMO dataset including Boxes, Floor, Wall, Table, and Fast are used. Both MOD and EV-IMO datasets were recorded using the same event cameras, namely DAVIS 346c which generates events within a spatial resolution of  $346 \times 260$  pixels.

EV-IMO2 [32] is a second version of EV-IMO, recorded using a higher resolution event camera ( $480 \times 640$  pixels). EV-IMO2 contains sequences of small-to-large and slow-tofast moving objects with various illumination conditions. It is targeted for motion segmentation approaches, depth estimation, optical flow, visual odometry, and SLAM methods [32]. Finally, we release as part of this paper a new event dataset, called EMS-DOMEL. This dataset is recorded locally in our lab facilities and includes several scenes of one to four dynamic objects captured using a moving camera. Event sequences are labeled using the DOMEL approach, described in and labeled using the proposed Section III-C. EMS-DOMEL will be used exclusively for testing. None of the sequences will be exposed to GTNN during the training phase and hence, the results will demonstrate the model's ability to generalize across various environments.

To train the GTNN algorithm, supervised learning is performed using the backpropagation technique. Pytorch is used to construct all the neural networks for training and testing. The Adam optimizer with a learning rate of 0.001 is used to execute training process to minimize the Dual Focal Loss (DFL). DFL was proposed in [58] to tackle the challenge of imbalanced training datasets for neural classification.

## B. Evaluation Metrics

To quantitatively evaluate the performance of our proposed approach, four evaluation metrics are used including F1 score, *Recall* metric, *Intersection over Union* (*IoU*), and *Detection Rate* (*DR*). The first two metrics are commonly used for object detection algorithms [59]. Particularly, *Recall* measures the ratio of the correctly detected dynamic objects to the true number of dynamic objects in the scene, as defined in Eq. (4). Whereas *Precision* measures the the ratio of the correctly detected dynamic objects to the true function objects to the total positive detection, as defined in Eq.(5). *F1* score, in consequence, computes the harmonic mean of *Precision* and *Recall*, as defined in Eq. (6).

$$Recall = \frac{TP}{TP + FN},\tag{4}$$

$$Precision = \frac{TP}{TP + FP},\tag{5}$$

$$F1 = 2 * \frac{Recall * Precision}{Recall + Precision},$$
(6)

where TP, FP, TN, and FN are the number of true positives, false positives, true negatives, and false negatives pixels, respectively. TP and FP indicate the number of events that are correctly and incorrectly predicted as dynamics objectsrelated events, respectively. While TN and FN indicate the number of events that are correctly and incorrectly predicted as background-related events. *Recall* and F1 are used to assess the capability of the proposed algorithm to distinguish between foreground and background events during the ablation study as discussed in the supplementary material.

Detection Rate (DR) and Intersection over Union (IoU), are two standard metrics used to quantitatively evaluate the performance of SOTA motion segmentation models as reported in [26, 33, 25]. DR is the area overlap between the bounding

box containing the predicted foreground events and the corresponding ground truth, as introduced in [33, 25]. The detection is considered successful if the following criteria is met:

Success if: 
$$(D \cap G) > 0.5$$
 and  $(D \cap G) > (D \cap G^c)$ , (7)

where D is the predicted bounding box (or convex hull), G is the corresponding ground truth bounding box (or convex hull), and  $G^c$  denotes the complement of the G set.

Intersection over Union (IoU) is the most commonly used evaluation metric for segmentation algorithms and has been used for event-based algorithms as reported in [26]. IoU is defined in the following equation:

$$IoU = \frac{S_D \cap S_G}{S_D \cup S_G},\tag{8}$$

where  $S_D$  is refers to the predicted object(s) segmentation mask and  $S_G$  is the corresponding ground truth mask. To generate a dense segmentation mask,  $S_D$  and  $S_G$ , the events are first projected into the 2D frame captured within the specified time window. Then, the convex hull outlining the events that belong to the dynamic object is computed. Consequently, all pixels within that convex hull are considered to be foreground events.

#### C. Quantitative Evaluation

In this section, we first provide a comparison between the performance of our GTNN model against SOTA learningbased motion segmentation model, SpikeMS as proposed in [26], in terms of IoU%.

Motion segmentation is carried out on the evaluation sequences from EV-IMO and MOD datasets, which were not exposed to the network during training. The results obtained using GTNN and SpikeMS [26] are reported in Table III. In addition, the IoU% metric obtained using GTNN for EV-IMO2 evaluation dataset is 49.76%. The segmentation results, based on the IoU% metric, obtained using GTNN outperform SpikeMS [26] in almost all sequences with an average of 9.4%. This is clearly due to the fact that graph transformers with the attention mechanism reveal and exploit the spatiotemporal correlations between the events that belong to the moving object and segregate them from background events. The network structure also aids extraction of global feature information and aggregates them with the local features extracted by the graph transformer's encoder-decoder units. Our network is able to distinguish between foreground and background events, especially when the relative speed of the camera and the moving objects is distinct.

Additionally, the performance of the proposed GTNN model is compared to classical hand-crafted motion segmentation methods [33, 25] and SpikeMS [26] based on the detection rate (DR%) metric, as reported in Table IV. Note that our GTNN model performs event-based motion segmentation without any prior knowledge about the scene geometry, the number of dynamic objects within the scene, and without any initialization stage, as opposed to the SOTA classical models [33, 25]. The results of the SOTA models are taken directly from their corresponding publications [33, 25, 26]. Our approach outperforms SpikeMS [26], the learning-based model, and has comparable performance against offline classical-based approaches [33, 25]. It can be concluded that the effectiveness of the proposed GTNN algorithm is validated across multiple unseen event streams acquired by a moving camera in a dynamic environment. The proposed algorithm could be integrated with other vision-based modules to ensure safe robotic navigation in an unknown and dynamic environment under challenging illumination conditions.

### D. Qualitative Evaluation

In this section, the testing sequences from the publiclyavailable datasets, MOD, EV-IMO and EV-IMO2, and our recorded experiments are used to qualitatively evaluate the motion segmentation performance. Note that, the tested sequences are not exposed to our network during training and the output of GTNN has the same structure as the 3D graph input. However, for visualization purposes, the results are projected into 2D frames. The results obtained from the proposed model are projected against the approximate ground truth and SOTA models SpikeMS [26] and Zhou et al. [24].

# 1) EV-IMO dataset

a) GTNN vs. learning-based approach (SpikeMS)

In this section, the performance of GTNN will be qualitatively compared to SpikeMS on the EV-IMO dataset. Fig. 8 shows selected scenes from three different sequences; (1) Floor, (2) Fast, and (3) Wall, from the EV-IMO dataset. Every row shows the 2D frame, the corresponding ground truth events, GTNN segmentation results, and SpikeMS segmentation results for the selected scenes, respectively. It is worth noting that SpikeMS [26] segmentation results are obtained using their publicly available pretrained model. For a fair comparison, a 10ms time window is selected to slice and prepare the testing samples for network prediction, since it is the time window selected to train SpikeMS [26].

Fig. 8-(A) depicts a sample scene from the Floor scenario, where a toy plane is moving above a highly textured carpet. The foreground events detected by GTNN solely belong to the moving object, while SpikeMS falsely segmented some of the background events as foreground events. GTNN was able to accurately segment the moving object in presence of significant background variations and motion dynamics, compared to SpikeMS.

Fig. 8-(B) shows a another sample scene captured in the Fast scenario, where the camera is experiencing harsh motion dynamics (rotation and translation at a high speed) while having a dynamic object within the scene. Our proposed model was able to distinguish events that correspond to dynamic objects in the scene from the background. Conversely, SpikeMS [26] has falsely segmented scattered background events, as foreground events.

Furthermore, in Fig. 8-(C), the performance of the GTNN model and SpikeMS [26] are tested on the Wall scenario, where multiple moving objects with different sizes, are moving along different random trajectories. Our segmentation results show better interpretation and fine labeling of the dynamic objects compared to SpikeMS [26]. This proves the robustness and generalization capability of our model to different environments with significant background textures and motion dynamics. Accurate motion segmentation results are impor-

TABLE III: Performance of the GTNN algorithm compared to SOTA motion segmentation learning-based method, SpikeMS [26] on EV-IMO and MOD event-based datasets. Segmentation results are compared using the IoU (in %)  $\uparrow$  metric.

	EV-IMO dataset										MOD datasat							
Method	Boxes		Floor		Wall		Table		Fast		wioD dataset		ει					
	100 ms	20 ms	10ms	100 ms	20 ms	10ms	100 ms	$20\ \mathrm{ms}$	10ms	100 ms	20 ms	10 ms	100 ms	20 ms	10 ms	100 ms	20 ms	10ms
SpilesMS <sup>†</sup> [26]	61.7	6519		60+5	52+16		65±7	6216		52+12	5019		45 1 1 1	28+10		69.7	65±5	
Spikewis' [20]	0111/	0510	-	00±3	JJ±10	-	05±7	05±0	-	32±13	30±0	-	45±11	30±10	-	08±/	05±5	-
GTNN model (ours)	49±36	40±31	34±30	77±13	68±19	63±22	77±15	65±21	58±23	71±23	62±27	- 56±29	$45 \pm 11$ 65±22	38±10 49±25	- 43±25	78±21	03±3 73±30	73±31

TABLE IV: Comparison with SOTA *learning-based* and *classical* approaches on sequences from EV-IMO and MOD event-based datasets, in terms of *Detection Rate (DR)* (in %) metric. Note that "C" and "L" indicate the type of the adopted approach; classical or learning, respectively.

Method	Туре	EV-1	IMO da	taset	MOD dataset			
Wiethod		100ms	20ms	10ms	100ms	20ms	10ms	
Mitrokhin et al. <sup>†</sup> [33]	C		48.79			70.12		
0-MMS <sup>‡</sup> [25]	С	81.06			82.35			
SpikeMS. <sup>†</sup> [26]	L	65.14			68.82			
Ours (GTNN model)	L	73	56	48	84	77	75	



(C) EV-IMO - Wall

Fig. 8: Segmentation results compared with the SOTA learning-based motion segmentation model (SpikeMS [26]) on testing sequences from EV-IMO dataset.

tant for various applications including event-based instance segmentation [60], recognition [61], and localization [62, 2] modules.

#### b) GTNN vs. classical-based approach

To further verify the validity of our proposed model, we analyze the performance of GTNN on other unseen testing scenarios of EV-IMO dataset, particularly from the Boxes and Table sequences, and show qualitatively the results compared to Zhou et al. [24] model. Note that Zhou et al. [24] model is a recent classical-based approach where motion segmentation is solved as an optimization problem of classical multi-model fitting schemes.

Although Zhou et al. [24] is an offline approach that undergoes an initialization stage based on motion compensation, the proposed GTNN segmentation results are highly accurate and comparable with Zhou et al. [24] as depicted in Fig. 9. In Boxes scene, one dynamic object, a toy car, traverses a textured carpet from left to right and we continuously segment the moving objects along the path, as shown in Fig. 9-(A). On the other hand, in the Table scene, two dynamic objects, a toy



(B) EV-IMO - Table seq01 - two dynamic objects within the scene

Fig. 9: Qualitative comparison with classical approach for eventbased motion segmentation on EV-IMO testing sequences (Boxes and Table). Each sample presents (left to right) APS frames (for visualization only), raw event stream, Zhou et al. [24] segmentation labels, and our GTNN prediction, respectively.



(C) EV-IMO2 - Evaluation Set - seq\_15\_02 (D) Ours (EMS-DOMEL) dataset - Seq1

Fig. 10: Qualitative evaluation of our GTNN algorithm on unseen testing simulation (MOD) and experimental (EV-IMO2 and our EMS-DOMEL) datasets. Our GTNN motion segmentation algorithm is able to identify dynamic objects within a variety of scenes and from various event camera resolutions ( $346 \times 260$  and  $640 \times 480$ ), when the camera is in motion.

plane and a car, move towards each other and then collide in the middle above the carpet, as shown in Fig. 9-(B). When the dynamic objects collided, our model segmented them together as one object. Segmentation results have proved the robustness of the proposed algorithm in presence of background variations and multiple dynamic objects.

#### 2) MOD, EV-IMO2 and our datasets (EMS-DOMEL)

Fig. 10 provides additional qualitative results of our approach on the simulated MOD dataset and experimental EV- IMO2 dataset compared against approximate event labels estimated from the mask of dynamic objects, provided alongside the datasets. Note that these sequences are not fed to our GTNN model during training. As shown in Fig. 10-(A), the approximate ground truth has some false labeling of the true dynamic objects (i.e. edges), however, our GTNN model predicts correctly the labels of the detected dynamic objects including their edges. Although, our model is trained with these approximate labels, i.e. including some false labels, the trained GTNN is able to exploit the spatiotemporal correlations of the events that belong to the moving object to correctly segment them from the background events. Fig. 10-(B-C) presents the predictions of GTNN model when tested on scenes from the EV-IMO2 dataset. Our segmentation shows good performance compared to the ground truth.

Further testing and analysis are done on our EMS-DOMEL recorded experimental dataset without any further training or fine-tuning of GTNN. GTNN demonstrated good segmentation and generalization capabilities. Lastly, we provide our segmentation results of a scenario where two toy cars with different sizes move from right to left and then collide with each other (EMS-DOMEL-Seq1). GTNN predictions continuously detect the moving objects (cars) while the camera is in motion, as shown in Fig. 10-(D). Additional segmentation results can be found at <https://youtu.be/9z3Ik8V45Ms> and are also provided in the supplementary material. To that end, the presented qualitative analyses have proven the capability of the proposed GTNN model to (1) cope with different camera parameters and various camera resolutions, and (2) generalize well to various dynamic scenes with multiple dynamic objects (moving at different speeds and in different directions) while the camera is in motion.

## E. GTNN Transferability Across Different Domains

GTNN is engineered in a way to facilitate its adaptability and generalizability, enabling its application across various domains and modalities. To tailor it for specific tasks, further refinement is often necessary. For instance, Sanket et al. [63] employed GTNN in the domain of event-based panoptic segmentation, by means of a transfer learning approach. They fed a 3D event stream graph into the GTNN, facilitating the exchange and nonlinear transformation of features to extract both local and global spatiotemporal relationships among graph nodes. The processed information is then directed to a final layer, which is specifically adjusted for the panoptic segmentation task. Hence, training was conducted for the final layer only, leaving the parameters of the early layers in the GTNN architecture unchanged, as demonstrated in [63].

Beyond vision-related tasks, GTNN's adaptability extends to non-visual tasks as well. This is intrinsic to its design framework, and is achieved by modifying the nodes and edges within the 3D graph structure to encapsulate the unique aspects of the problem being addressed. For instance, in nonvision-related tasks, the graph elements could be redefined to represent different data types or relationships, showcasing the GTNN's broad generalization capabilities. This adaptability underscores the potential of GTNN to serve as a base architecture for a wide range of applications, provided that appropriate adjustments are made to align with the specific requirements of each task.

# F. Limitations

## 1) Segmentation Challenges in Dynamic Environments

Rigorous evaluations are conducted on the publiclyavailable datasets as well as our recorded experiments demonstrating the superiority of the GTNN to segment dynamic objects (Sections IV-C and IV-D). Our evaluations indicated that GTNN struggles to differentiate dynamic objects from the background in scenarios where the camera and object speeds are similar, often misclassifying all events as background. This limitation becomes more pronounced in environments with minimal relative motion between the camera and the object, leading to inaccurate foreground-background segmentation. Furthermore, GTNN encounters difficulties during rapid camera rotations, especially in situations where the scene is cluttered with numerous background events. In these situations, the model's effectiveness in recognizing and segregating smaller or distant moving objects is reduced. Rapid camera movements exacerbate the problem by flooding the algorithm with an excessive event stream. This issue is evidenced by our observed reduction in performance for the 'EV-IMO Boxes' scenarios, as detailed in Table III. Such scenarios pose a substantial challenge for GTNN in consistently recognizing and tracking objects. It is worth mentioning that our GTNN yet demonstrates outstanding performance in some 'EV-IMO Boxes' evaluation sequences, as presented in Fig. 9.

2) Computational Time Analysis

In this section, analysis of computational time required to process input event streams by the proposed GTNN algorithm is presented and compared to SpikeMS [26]. Note that timing analysis was carried out on a Dell Desktop Computer with Intel(R) Xeon(R) W-2145@2.70GHz×8 and two Nvidia Quadro RTX 6000 GPUs. A set of 30 event graphs, each spanning a 10ms time window was selected from the Boxes sequence of EVI-MO dataset to conduct the time analysis. The computational time analysis of the proposed algorithm compared to SpikeMS [26] was done in terms of the forward prediction time as shown in Fig. 11. It is observed that the time needed to process each event graph was shorter using our proposed approach than SpikeMS [26] achieving a double speedup of processing time.

There is yet more room for improvement to expedite the runtime performance of the proposed approach to fulfill the real-time performance requirement and achieve the primary goal of the event-based motion segmentation. Therefore, to achieve real-time performance, processing a group of events within a selected time window should be performed before receiving the next event graph, i.e. in our case processing should be done in less than 10ms. This could be achieved by optimizing the complexities of the GTNN architecture via knowledge distillation methods [64]. More particularly, a reduced network architecture (called a student model) is trained in parallel with the complex network (called a teacher model) which has enough nonlinear capacity to handle the problem at hand. The student model, when trained alone, does not have the capability to map the nonlinearities in the input and



Fig. 11: Forward time in seconds to perform motion segmentation for 10ms time window of event stream using the proposed approach and SpikeMS [26]

output dataset to provide the required performance, however, with the student-teacher parallel training methods, this can be achieved. Alternatively, to reduce the runtime, a simpler network than the proposed GTNN architecture is suggested to be trained using knowledge distillation framework. These areas for improvement will be addressed in the future.

## V. CONCLUSIONS

In this work, we presented the first learning-based graph transformer neural network (GTNN) algorithm tackling a large-scale problem in computer vision using dynamic vision sensors. Our GTNN algorithm is developed to infer spatiotemporal patterns of the acquired event streams and perform eventbased motion segmentation accordingly. More specifically, the algorithm reveals the motion dynamics of the camera and the scene and decides whether the incoming events represent foreground (due to moving objects) or background (due to camera motion) event data.

The proposed GTNN successfully operates on event streams without any initialization stage nor prior knowledge in terms of scene geometry, motion patterns, or number of independent dynamic objects. This is attributed to the fact that the adopted graph structure of the input data exploits the spatiotemporal patterns between the events, then infers them in the global context of the scene. Event streams are processed in their asynchronous form which preserves their temporal attributes, i.e. without projecting them into 2D frames. Such operation is carried out in the point transformer with self-attention mechanism, transition up and transition down layers where inputs are fed as 3D event graphs which could be of variable sizes.

GTNN algorithm was trained using the proposed effective scheme on three publicly available synthetic and real-work sequences, MOD, EV-IMO, and EV-IMO2. The effective training scheme uses a portion of the extensive training datasets at every iteration (epoch) for tuning the network weights. The proposed effective scheme has reduced the training time and expedited convergence. The proposed GTNN algorithm has outperformed the SOTA learning-based motion segmentation [26] and classical methods [24, 33] in all the testing scenarios with at least 4.5% higher detection rate and 9.5% accuracy (IoU%) on testing sets. Moreover, the forward prediction time is 50% less compared to SOTA learning-based approach. Furthermore, qualitative results have proven the superiority of the proposed algorithm to both the learning-based motion segmentation approach [26] and the classical approach proposed in [24].

Our GTNN model was also tested on our experimental dataset which was not exposed to the network during training. A novel offline event labeling technique, referred to as DOMEL, is proposed to label the recorded dataset, which involves event streams capturing moving objects when the camera is in motion. GTNN is able to successfully segment the moving objects from the background, despite the fact that the data is recorded under conditions different than those of the training data; a variety of scenes, multiple objects moving with various speeds and in random paths, and unknown camera motions with various sensor resolutions. This work in addition to our previous research [29] have unlocked the potential of using graph transformers neural networks on vision-based navigation modules with event camera. In the future, we plan to demonstrate the significance of our proposed eventbased motion segmentation algorithm by integrating it with a dynamic object avoidance module to perform safe navigation in unknown dynamic environments.

#### ACKNOWLEDGEMENTS

This research publication was funded by the Khalifa University of Science and Technology under Award No. RC1-2018-KUCARS, and by the Advanced Research and Innovation Center (ARIC), which is jointly funded by Khalifa University of Science and Technology and STRATA Manufacturing PJSC (a Mubadala company), grant number 8436010. The author(s) wish to acknowledge the contribution of Khalifa University's high-performance computing and research computing facilities to the results of this research. The author(s) also thank Mohammed Salah and Oussama Abdul Hay for their help during data collection.

## REFERENCES

- [1] G. N. DeSouza and A. C. Kak, "Vision for mobile robot navigation: A survey," *IEEE TPAMI*, vol. 24, no. 2, pp. 237–267, 2002.
- [2] Y. Alkendi, L. Seneviratne, and Y. Zweiri, "State of the art in visionbased localization techniques for autonomous navigation systems," *IEEE Access*, vol. 9, pp. 76847–76874, 2021.
- [3] I. Lluvia, E. Lazkano, and A. Ansuategi, "Active mapping and robot exploration: A survey," *Sensors*, vol. 21, no. 7, p. 2445, 2021.
- [4] H.-J. Liang, N. J. Sanket, C. Fermüller, and Y. Aloimonos, "Salientdso: Bringing attention to direct sparse odometry," *IEEE TASE*, vol. 16, no. 4, pp. 1619–1626, 2019.
- [5] T. Rateke and A. von Wangenheim, "Passive vision road obstacle detection: a literature mapping," *International Journal of Computers and Applications*, vol. 44, no. 4, pp. 376–395, 2022.
- [6] A. Beghdadi and M. Mallem, "A comprehensive overview of dynamic visual slam and deep learning: concepts, methods and challenges," *Machine Vision and Applications*, vol. 33, no. 4, pp. 1–28, 2022.
- [7] Y. Bi, B. Xue, P. Mesejo, S. Cagnoni, and M. Zhang, "A survey on evolutionary computation for computer vision and image analysis: Past, present, and future trends," *IEEE Transactions on Evolutionary Computation*, 2022.
- [8] B. Emek Soylu, M. S. Guzel, G. E. Bostanci, F. Ekinci, T. Asuroglu, and K. Acici, "Deep-learning-based approaches for semantic segmentation of natural scene images: A review," *Electronics*, vol. 12, no. 12, 2023. [Online]. Available: https://www.mdpi.com/2079-9292/12/12/2730
- [9] W. Wang, X. Lu, J. Shen, D. J. Crandall, and L. Shao, "Zero-shot video object segmentation via attentive graph neural networks," in *Proceedings* of the IEEE/CVF international conference on computer vision, 2019, pp. 9236–9245.
- [10] X. Lu, W. Wang, M. Danelljan, T. Zhou, J. Shen, and L. Van Gool, "Video object segmentation with episodic graph memory networks," in *Computer Vision–ECCV 2020: 16th European Conference, Glasgow*,

UK, August 23–28, 2020, Proceedings, Part III 16. Springer, 2020, pp. 661–679.

- [11] J. Mattheus, H. Grobler, and A. M. Abu-Mahfouz, "A review of motion segmentation: Approaches and major challenges," in 2020 2nd International Multidisciplinary Information Technology and Engineering Conference (IMITEC), 2020, pp. 1–8.
- [12] T. Zhou, F. Porikli, D. J. Crandall, L. Van Gool, and W. Wang, "A survey on deep learning technique for video segmentation," *IEEE Transactions* on Pattern Analysis and Machine Intelligence, vol. 45, no. 6, pp. 7099– 7122, 2022.
- [13] J. Mattheus, H. Grobler, and A. M. Abu-Mahfouz, "A review of motion segmentation: Approaches and major challenges," in 2020 2nd IMITEC. IEEE, 2020, pp. 1–8.
- [14] K. Jo, S. Lee, C. Kim, and M. Sunwoo, "Rapid motion segmentation of lidar point cloud based on a combination of probabilistic and evidential approaches for intelligent vehicles," *Sensors*, vol. 19, no. 19, p. 4116, 2019.
- [15] A. Mitrokhin, C. Ye, C. Fermüller, Y. Aloimonos, and T. Delbruck, "Ev-imo: Motion segmentation dataset and learning pipeline for event cameras," in 2019 IEEE/RSJ IROS, 2019, pp. 6105–6112.
- [16] G. Gallego, T. Delbruck, G. M. Orchard, C. Bartolozzi, B. Taba, A. Censi, S. Leutenegger, A. Davison, J. Conradt, K. Daniilidis, and D. Scaramuzza, "Event-based vision: A survey," *IEEE TPAMI*, pp. 1–1, 2020.
- [17] A. R. Vidal, H. Rebecq, T. Horstschaefer, and D. Scaramuzza, "Ultimate slam? combining events, images, and IMU for robust visual SLAM in HDR and high-speed scenarios," *IEEE RAL*, vol. 3, no. 2, pp. 994–1001, 2018.
- [18] M. Salah, M. Chehadah, M. Humais, M. Wahbah, A. Ayyad, R. Azzam, L. Seneviratne, and Y. Zweiri, "A neuromorphic vision-based measurement for robust relative localization in future space exploration missions," *IEEE TIM*, 2022.
- [19] J. Furmonas, J. Liobe, and V. Barzdenas, "Analytical review of eventbased camera depth estimation methods and systems," *Sensors*, vol. 22, no. 3, p. 1201, 2022.
- [20] R. Jiang, X. Mou, S. Shi, Y. Zhou, Q. Wang, M. Dong, and S. Chen, "Object tracking on event cameras with offline-online learning," *CAAI Transactions on Intelligence Technology*, vol. 5, no. 3, pp. 165–171, 2020.
- [21] A. Ayyad, M. Halwani, D. Swart, R. Muthusamy, F. Almaskari, and Y. Zweiri, "Neuromorphic vision based control for the precise positioning of robotic drilling systems," *Robotics and Computer-Integrated Manufacturing*, vol. 79, p. 102419, 2023.
- [22] F. L. Macdonald, N. F. Lepora, J. Conradt, and B. Ward-Cherrier, "Neuromorphic tactile edge orientation classification in an unsupervised spiking neural network," *Sensors*, vol. 22, no. 18, p. 6998, 2022.
- [23] T. Stoffregen, G. Gallego, T. Drummond, L. Kleeman, and D. Scaramuzza, "Event-based motion segmentation by motion compensation," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 7244–7253.
- [24] Y. Zhou, G. Gallego, X. Lu, S. Liu, and S. Shen, "Event-based motion segmentation with spatio-temporal graph cuts," *IEEE TNNLS*, 2021.
- [25] C. M. Parameshwara, N. J. Sanket, C. D. Singh, C. Fermüller, and Y. Aloimonos, "0-mms: Zero-shot multi-motion segmentation with a monocular event camera," in 2021 IEEE ICRA, 2021, pp. 9594–9600.
- [26] C. M. Parameshwara, S. Li, C. Fermüller, N. J. Sanket, M. S. Evanusa, and Y. Aloimonos, "Spikems: Deep spiking neural network for motion segmentation," in 2021 IEEE/RSJ IROS, 2021, pp. 3414–3420.
- [27] A. Mitrokhin, Z. Hua, C. Fermuller, and Y. Aloimonos, "Learning visual motion segmentation using event surfaces," in *Proceedings of the IEEE/CVF Conference on CVPR*, 2020, pp. 14414–14423.
- [28] H. Zhao, L. Jiang, J. Jia, P. H. Torr, and V. Koltun, "Point transformer," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 16259–16268.
- [29] Y. Alkendi, R. Azzam, A. Ayyad, S. Javed, L. Seneviratne, and Y. Zweiri, "Neuromorphic camera denoising using graph neural network-driven transformers," *IEEE TNNLS*, pp. 1–15, 2022.
- [30] M. Gehrig and D. Scaramuzza, "Recurrent vision transformers for object detection with event cameras," arXiv preprint arXiv:2212.05598, 2022.
- [31] A. Sabater, L. Montesano, and A. C. Murillo, "Event transformer. a sparse-aware solution for efficient event data processing," in *Proceedings* of the IEEE/CVF Conference on CVPR, 2022, pp. 2677–2686.
- [32] L. Burner, A. Mitrokhin, C. Fermüller, and Y. Aloimonos, "Evimo2: An event camera dataset for motion segmentation, optical flow, structure from motion, and visual inertial odometry in indoor scenes with monocular or stereo algorithms," *arXiv preprint arXiv:2205.03467*, 2022.

- [33] A. Mitrokhin, C. Fermuller, C. Parameshwara, and Y. Aloimonos, "Event-based moving object detection and tracking," 2018 IEEE/RSJ IROS, Oct 2018. [Online]. Available: http://dx.doi.org/10.1109/IROS. 2018.8593805
- [34] F. Barranco, C. Fermuller, and E. Ros, "Real-time clustering and multitarget tracking using event-based sensors," in 2018 IEEE/RSJ IROS. IEEE, 2018, pp. 5764–5769.
- [35] A. Linares-Barranco, F. Gómez-Rodríguez, V. Villanueva, L. Longinotti, and T. Delbrück, "A usb3.0 fpga event-based filtering and tracking framework for dynamic vision sensors," in 2015 IEEE ISCAS, 2015, pp. 2417–2420.
- [36] A. Mishra, R. Ghosh, J. C. Principe, N. V. Thakor, and S. L. Kukreja, "A Saccade Based Framework for Real-Time Motion Segmentation Using Event Based Vision Sensors," *Frontiers in Neuroscience*, vol. 11, p. 83, 2017. [Online]. Available: https://www.frontiersin.org/article/10.3389/fnins.2017.00083
- [37] A. Ogale, C. Fermuller, and Y. Aloimonos, "Motion segmentation using occlusions," *IEEE TPAMI*, vol. 27, no. 06, pp. 988–992, jun 2005.
- [38] G. Gallego, M. Gehrig, and D. Scaramuzza, "Focus is all you need: Loss functions for event-based vision," in *Proceedings of the IEEE/CVF Conference on CVPR*, 2019, pp. 12280–12289.
- [39] T. Stoffregen and L. Kleeman, "Event cameras, contrast maximization and reward functions: An analysis," in 2019 IEEE/CVF Conference on CVPR (CVPR), 2019, pp. 12 292–12 300.
- [40] S. Xiao, S. Wang, Y. Dai, and W. Guo, "Graph neural networks in node classification: survey and evaluation," *Machine Vision and Applications*, vol. 33, no. 1, pp. 1–19, 2022.
- [41] J. H. Giraldo, S. Javed, and T. Bouwmans, "Graph moving object segmentation," *IEEE TPAMI*, 2020.
- [42] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in neural information processing systems*, 2017, pp. 5998–6008.
- [43] K. Han, Y. Wang, H. Chen, X. Chen, J. Guo, Z. Liu, Y. Tang, A. Xiao, C. Xu, Y. Xu et al., "A survey on visual transformer," arXiv preprint arXiv:2012.12556, vol. 2, no. 4, 2020.
- [44] D. Lin, Y. Li, Y. Cheng, S. Prasad, A. Guo, and Y. Cao, "Multi-range view aggregation network with vision transformer feature fusion for 3d object retrieval," *IEEE Transactions on Multimedia*, pp. 1–12, 2023.
- [45] J. Jiao, Y.-M. Tang, K.-Y. Lin, Y. Gao, J. Ma, Y. Wang, and W.-S. Zheng, "Dilateformer: Multi-scale dilated transformer for visual recognition," *IEEE Transactions on Multimedia*, pp. 1–14, 2023.
- [46] X. Lin, S. Sun, W. Huang, B. Sheng, P. Li, and D. D. Feng, "Eapt: Efficient attention pyramid transformer for image processing," *IEEE Transactions on Multimedia*, vol. 25, pp. 50–61, 2023.
- [47] Q. Xu, J. Wang, B. Jiang, and B. Luo, "Fine-grained visual classification via internal ensemble learning transformer," *IEEE Transactions on Multimedia*, pp. 1–14, 2023.
- [48] M.-H. Guo, J.-X. Cai, Z.-N. Liu, T.-J. Mu, R. R. Martin, and S.-M. Hu, "Pct: Point cloud transformer," *Computational Visual Media*, vol. 7, no. 2, pp. 187–199, 2021.
- [49] X. Yu, L. Tang, Y. Rao, T. Huang, J. Zhou, and J. Lu, "Point-bert: Pretraining 3d point cloud transformers with masked point modeling," in *Proceedings of the IEEE/CVF Conference on CVPR*, 2022, pp. 19313– 19322.
- [50] X.-F. Han, Y.-F. Jin, H.-X. Cheng, and G.-Q. Xiao, "Dual transformer for point cloud analysis," *IEEE Transactions on Multimedia*, 2022.
- [51] H. Zhao, J. Jia, and V. Koltun, "Exploring self-attention for image recognition," in *Proceedings of the IEEE/CVF Conference on CVPR*, 2020, pp. 10076–10085.
- [52] B. Pu, H. Kim, X.-D. Cai, B. Sen, C. Sui, and J. Fan, "Training set optimization in an artificial neural network constructed for high bandwidth interconnects design," *IEEE T-MTT*, vol. 70, no. 6, pp. 2955– 2964, 2022.
- [53] I. Valova, C. Harris, T. Mai, and N. Gueorguieva, "Optimization of convolutional neural networks for imbalanced set classification," *Procedia Computer Science*, vol. 176, pp. 660–669, 2020, knowledge-Based and Intelligent Information & Engineering Systems: Proceedings of the 24th International Conference KES2020.
- [54] "Robot operating system." [Online]. Available: https://www.ros.org/
- [55] P. Bergström and O. Edlund, "Robust registration of point sets using iteratively reweighted least squares," *Computational optimization and applications*, vol. 58, no. 3, pp. 543–561, 2014.
- [56] J. Canny, "A computational approach to edge detection," *IEEE TPAMI*, no. 6, pp. 679–698, 1986.
- [57] "Image segmenter app." [Online]. Available: https://www.mathworks. com/help/images/ref/imagesegmenter-app.html

- [58] M. S. Hossain, J. M. Betts, and A. P. Paplinski, "Dual focal loss to address class imbalance in semantic segmentation," *Neurocomputing*, vol. 462, pp. 69–87, 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0925231221011310
- [59] J. Kaur and W. Singh, "Tools, techniques, datasets and application areas for object detection in an image: a review," *Multimedia Tools and Applications*, pp. 1–55, 2022.
- [60] P. Kirkland, D. Manna, A. Vicente, and G. Di Caterina, "Unsupervised spiking instance segmentation on event data using stdp features," *IEEE Transactions on Computers*, vol. 71, no. 11, pp. 2728–2739, 2022.
- Transactions on Computers, vol. 71, no. 11, pp. 2728–2739, 2022.
  [61] C. Liu, X. Qi, E. Y. Lam, and N. Wong, "Fast classification and action recognition with event-based imaging," *IEEE Access*, vol. 10, pp. 55 638–55 649, 2022.
- [62] J. Hidalgo-Carrió, G. Gallego, and D. Scaramuzza, "Event-aided direct sparse odometry," in *Proceedings of the IEEE/CVF Conference on CVPR*, 2022, pp. 5781–5790.
- [63] S. Kachole, Y. Alkendi, F. B. Naeini, D. Makris, and Y. Zweiri, "Asynchronous events-based panoptic segmentation using graph mixer neural network," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 4082–4091.
- [64] L. Wang and K.-J. Yoon, "Knowledge distillation and student-teacher learning for visual intelligence: A review and new outlooks," *IEEE TPAMI*, 2021.