Asynchronous Events-based Panoptic Segmentation using Graph Mixer Neural Network

Sanket Kachole[1] Yusra Alkendi[2] Fariborz Baghaei Naeini[1,3] Dimitrios Makris[1] Yahya Zweiri[2]
Dept. of Computer Science, Kingston University, London, UK[1] Ipsotek, an Eviden Company, London [3]
Advanced Research and Innovation Center (ARIC), Khalifa University, Abu Dhabi, UAE[2]
{K1742163,f.baghaeinaeini,d.makris}@kingston.ac.uk[1] {yusra.alkendi,y.zweiri}@ku.ac.ae[2]

Abstract

In the context of robotic grasping, object segmentation encounters several difficulties when faced with dynamic conditions such as real-time operation, occlusion, low lighting, motion blur, and object size variability. In response to these challenges, we propose the Graph Mixer Neural Network that includes a novel collaborative contextual mixing layer, applied to 3D event graphs formed on asynchronous events. The proposed layer is designed to spread spatiotemporal correlation within an event graph at four nearest neighbor levels parallelly. We evaluate the effectiveness of our proposed method on the Event-based Segmentation (ESD) Dataset, which includes five unique image degradation challenges, including occlusion, blur, brightness, trajectory, scale variance, and segmentation of known and unknown objects. The results show that our proposed approach outperforms state-of-the-art methods in terms of mean intersection over the union and pixel accuracy. Code available at: https://github.com/sanket0707/GNN-Mixer.git

1. Introduction

Object grasping is a crucial task for robots with applications in manufacturing, logistics, healthcare, and household tasks [10, 25]. However, detecting and segmenting objects accurately in the robot's environment is challenging due to occlusions, complex geometries, and dynamic backgrounds [7]. Panoptic segmentation aims to simultaneously segment foreground objects and background regions in an image. Integrating panoptic segmentation into object grasping enables robots to perceive their environment better and perform more complex tasks efficiently.

Common challenges in panoptic segmentation are due to cluttered scenes, object geometry and appearance variability [17, 30], occlusions, motion blur [18] and low temporal resolution [19] in traditional cameras. High latency can cause delays in processing sensor data, resulting in slower response times and reduced accuracy in performing tasks. Recent progress in object segmentation using State-of-the-Art Graph Neural Networks with a transformer mechanism [1, 2] introduces additional constraints, as both panoptic segmentation and grasp planning must be performed quickly and efficiently. To address these challenges, more sophisticated algorithms and techniques are needed to better handle the variability and complexity of real-world environments.

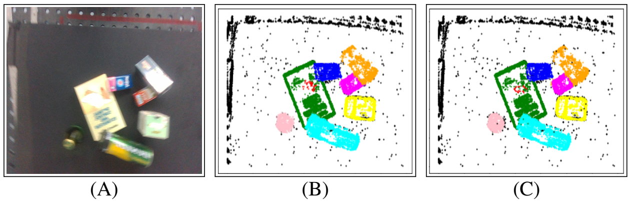

Figure 1. Panoptic segmentation results of the proposed learning-based algorithm (GMNN) applied on the ESD dataset [12]. (A) APS image for visualization only. (B) Ground truth approximate events (C) Segmented events using the GMNN algorithm.

We propose the Graph Mixer Neural Network (GMNN) for event-based panoptic segmentation. Our proposed model maintains the asynchronous nature of event streams and leverages spatiotemporal correlations to infer the scene. The key technical contribution is the novel Collaborative Contextual Mixing (CCM) layer within a graph neural network architecture that enables the parallel mixing of event features generated from multiple sets of neighborhood events. Our proposed model achieves state-of-the-art performance on the ESD dataset [12] which consists of robotic grasping scenes captured using an event camera mounted next to the gripper of a robotic arm. The dataset includes scenes with variations in object clutter size, arm speed, motion direction, distance between the object and camera, and lighting conditions. Specifically, it achieves superior results in terms of mean Intersection Over Union (mIoU) and pixel accuracy, while also demonstrating significant improvements in computational efficiency compared to existing state-of-the-art methods. Fig. 1 shows segmentation results obtained when testing our GMNN on a sample from the ESD dataset.

Previous work is discussed in Section II. Our proposed architecture is described in detail in Section III. The validation of our method through experimental results and an ablation study are presented in Section IV. Finally, the conclusion and scope for further research are outlined in Section V.

2. Related Work

2.1. Image segmentation Methods

Thresholding algorithms have fixed thresholds [20] and lack contextual information, while clustering techniques [2, 29] can adapt to variable structures but are sensitive to initial conditions and may lead to over-segmentation. Deep learning methods [23], [5], [15] produce dense predictions but may ignore small objects and details. Event-based methods have advantages as they can handle motion blur and high dynamic range [2] but may require labeled images such as the Event-based Semantic Segmentation (ESS) method [28] and have limitations in segmenting small objects [3]. Multiple modalities can be integrated to leverage complementarity, with CMX using transformer-based architecture [14] and Bimodal SegNet [13] fusing RGB with event frames. Although these methods demonstrate promising results, their limitations include overlooking the high temporal resolution of event-based data.

2.2. Graph neural network methods

The adoption and evolution of GNNs in computer vision applications has been remarkable in recent years [4]. Asynchronous Event-based GNNs (AEGNNs) [26] extend GNNs to process events as evolving spatiotemporal graphs. However, using conventional deep neural networks to process dense representations of events eliminates their sparsity and asynchronous nature, leading to computational and latency constraints. GNN-transformer [2] addresses the problem by utilizing spatiotemporally evolving graphs that can be efficiently and asynchronously processed using GNNs. TactileSGNet [11] utilized a spiking-GNN [27] and a GNN-Transformer algorithm to perform event-based recognition of tactile objects and classify active event pixels by leveraging the EventConv message-passing framework to capture spatiotemporal correlations among events while preserving their asynchronous nature [1].

Graph Transformer Neural Network (GTNN) applies a self-attention mechanism to motion segmentation in asynchronous event-based vision data streams using 3D graphs [2]. However, long-range dependencies pose a challenge for transformer-based models due to their spatially variant nature. Event-based Transformer (EvT) resolves this by creating event frames and employing sparse patch-based event data representation with attention mechanisms [24]. Despite the promise of transformer-based methods, modeling long-range dependencies is difficult due to their spatially variant nature. MLP-like architectures applied in 3-D point clouds outperform transformers and CNNs in handling position-sensitive information using simple token and channel-mixing MLPs [32], without self-attention mechanisms, on large-scale data. However, Metaformer shows that general architecture formulation is more critical than specific interaction strategies, achieving remarkable results by replacing token-mixing with average pooling [31]. The PointMixer method improves feature mixing within and between point sets using a universal point set operator instead of token-mixing MLPs, resulting in better parameter efficiency and accuracy with Softmax substitution [6].

MLPs excel in different applications but not in asynchronous event-based vision tasks, while GNNs have yet to explore modern MLP-like techniques [6, 32]. The current use of K-Nearest Neighbors (KNN) is insufficient to improve feature mixing within GNNs, and novel approaches are required for challenging tasks such as event-based panoptic segmentation.

3. Methodology

3.1. Prerequisite

3.1.1 Event-based vision data

Event-based vision cameras respond to changes in log intensities by capturing pixel-level changes called events. A continuous stream of events is mathematically represented by a sequence of tuples comprising ith event location (x_i, y_i), timestamp t_i, and polarity z_i [21, 22]:

$$(x_1, y_1, t_1, z_1), (x_2, y_2, t_2, z_2), ..., (x_n, y_n, t_n, z_n) \qquad (1)$$

3.1.2 Graph Neural Network

Graph Neural Networks (GNNs) consist of nodes or vertices V, connected by edges or links L [8]. Mathematically, a GNN can be expressed as $G = (V, L)$. Each node $i \in V$ takes as input the weighted sum of the output values q_i of its incoming edges o_i, and produces an output value $r_i = f_i(o_i^T q_i)$.

3.1.3 Multi-Layer Perceptron (MLP)

Each node $i \in V$ assumes an initial feature vector p_i^0 at layer 0, which is transformed using an MLP to produce new feature vectors p_i^l at each MLP layer l:

$$p_i^l = \sigma\left(W^l \sum_{j \in N_i} \frac{1}{c_{i,j}} p_j^{l-1} + b^l\right) \qquad (2)$$

where σ is a non-linear activation function, N_i denotes the set of neighboring nodes of queried node, W^l is weight matrix, b^l is bias vector of the MLP at layer l and $c_{i,j}$ is a normalization factor [6, 16].

3.1.4 K-Nearest Neighbors on Graph Nodes

The k-Nearest Neighbor (kNN) method is commonly used to process event data locally by considering proximity, resulting in an index map of neighboring nodes represented as M_i [9]. Let's assume a set of nodes $N = \{n_i\}_{i=1}^{N}$ with corresponding features $P = \{p_i\}_{i=1}^{N}$. For a query node n_i, an index map $M_{k,i}$ of the k closest nodes can be calculated using kNN as:

$$M_{k,i} = kNN(N, k, n_i) \tag{3}$$

The corresponding feature set for this kNN is $P_{k,i} = \{p_j \in P | j \in M_{k,i}\}$.

3.2. Graph Mixer Neural Network

In this section, the framework of the Graph Mixer Neural Network, depicted in Fig. 2, is explained in detail. The first section 3.2.1 describes a method for constructing a 3D graph to represent a series of events that occur within a predetermined time interval. The graph is constructed based on the most recent N_{max} events, and a k-Nearest Neighbor (kNN) search connects each node with its k-nearest neighboring nodes. The next section 3.2.2 the Collaborative Contextual Mixing (CCM) method, a novel approach for disseminating event features across various sets, is described. Further, in section 3.2.3 a transition down block to downsamples the graph, and in section 3.2.4 a transition up the block to upsample graph nodes are discussed.

3.2.1 3-D Graph Construction

A 3D graph G represents a series of events that occur within a pre-determined time interval T. Each node i represents an event in the stream, with features $e_i = [x_i, y_i, t_i]$. Event polarityvaries with camera parameter settings, and therefore limits the generalizability of the proposed algorithm. Therefore, we excluded polarity from the graph node features, following [6].

The number of events triggered by changes in pixel intensities in the output event stream depends on the camera's speed. Higher speed results in more events, while lower speed results in fewer events. These conditions make it difficult to determine spatiotemporal event relationships due to redundant data and high computational demands. In addition, a memory limit imposes a maximum number of events that can be stored, causing the potential loss of important information from earlier events. It also affects the ability to determine spatiotemporal relationships, resulting in reduced accuracy and precision. Therefore, the choice of N_{max} needs careful consideration to balance memory usage and accuracy. To address these issues and minimize memory consumption, the graph is constructed within each time interval T with a given maximum number of nodes N_{max}. If the number of events within this time window exceeds the N_{max}, e.g. because of the scene and camera dynamics, only the most recent N_{max} events are preserved. The method can be applied across different domains, regardless of the number of events triggered within the temporal window, thanks to the feature of graph-based neural networks to operate on graphs of varying sizes.

Spatiotemporal distances are scaled by dividing the spatial distances by the maximum spatial distances X and Y and the temporal distances by the maximum temporal distance T. This normalization ensures that the spatial and temporal distances are on the same scale and have equal weight in the calculation of the spatiotemporal distances. A k-Nearest Neighbor (kNN) search connects each node i with its corresponding feature p_i with its k-nearest neighboring nodes j and their corresponding features p_j using 3D normalized spatiotemporal distances. The resulting spatiotemporal neighborhoods are called sub-graphs, each with $k+1$ nodes. Once all subgraphs are constructed, each will pass through the nonlinear operations of the Mixer Layer where each node (event) features are encoded, and the Sampling Up/Down where graph nodes are convolved/deconvolved.

3.2.2 Collaborative contextual mixing

Feature mixing is one of the important aspects in graph neural networks to understand the relationships between nodes. Existing feature mixing methods employ kNN-based subsampling to disseminate the features. We argue that the use of k-nearest neighbors (kNN) solely is inadequate as it confines an event to collect information from a restricted neighborhood. Considering the sparse and asynchronous nature of events, the event-based vision domain demands advanced techniques.

We introduce the Collaborative Contextual Mixing (CCM) method as a novel approach to disseminating event features across various sets. Thus, distributing the event features among multiple levels of the nearest neighbors in parallel and then aggregating them using a weighted sum results in a more effective feature mixing as shown in Fig. 4.

First, a spatial pyramidal block of $kNNs$ is applied simultaneously at four levels with $k \in \{16, 32, 48, 64\}$ (kNN_1 to kNN_4) which produces four index maps $M_{k,i}$, each with a corresponding feature set $P_{k,i}$. This choice was made based on the network, in accordance with the ablation study conducted on its hyperparameters and variants. For each query node n_i, at level k, a score vector $s = [s_1, ...s_k]$

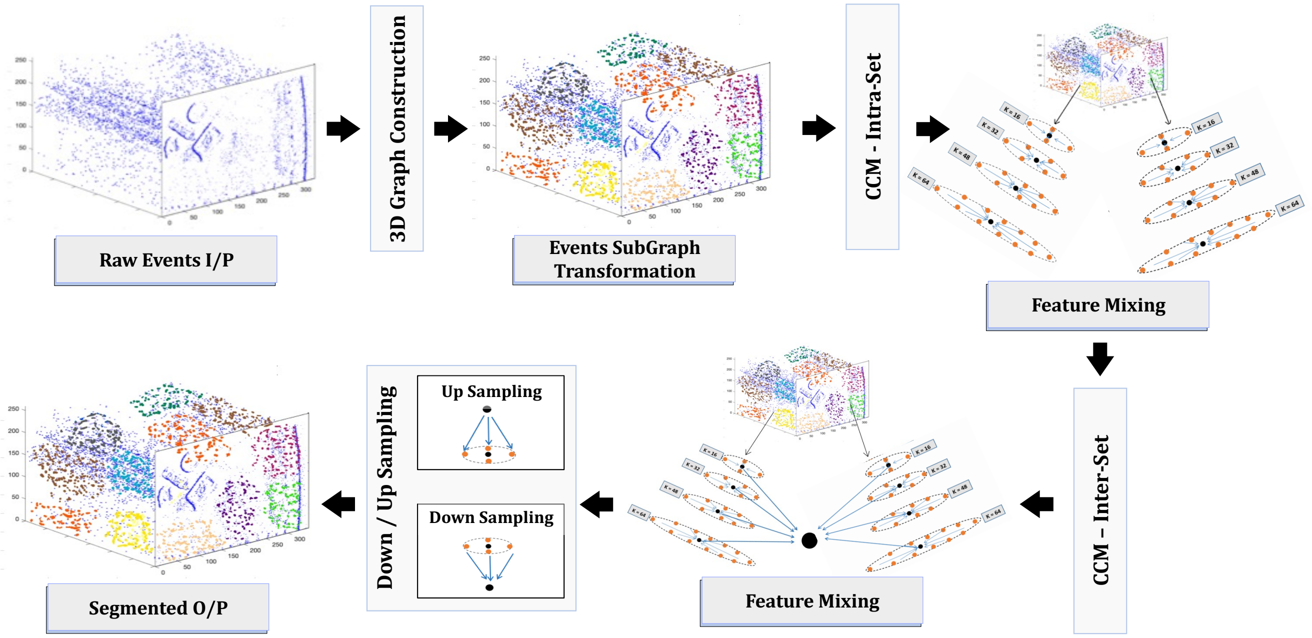

Figure 2. Proposed Framework - Graph Mixer Neural Network (GMNN) for panoptic segmentation of asynchronous event data in a robotic environment. GMNN operates on a 3D- graph constructed of DVS events acquired within a temporal window, encapsulating its spatiotemporal properties. Subgraphs of spatiotemporally neighboring events are then constructed (colored event in step 2) where each is processed by various nonlinear operations within Mixer and sampling modules to perform segmentation.

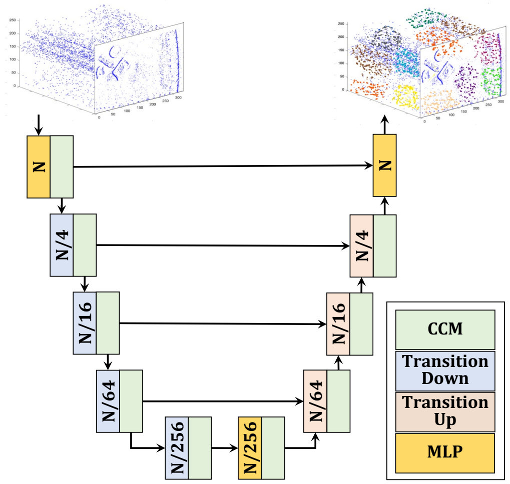

Figure 3. Graph Mixer Neural Network (GMNN). Note that "N" represents the number of nodes (i.e., Events) per graph.

is computed:

$$s_j = g_2([g_1(x_j); \delta(e_i - e_j)], where j \in M_{k,i} \qquad (4)$$

where g is a channel mixing MLP, δ refers to the relative positional encoding MLPs and p_j is a j-th element of the feature vector set P_i. The computed score vector is then passed into the following function to compute the output feature vector $u_{k,i}$ as follows:

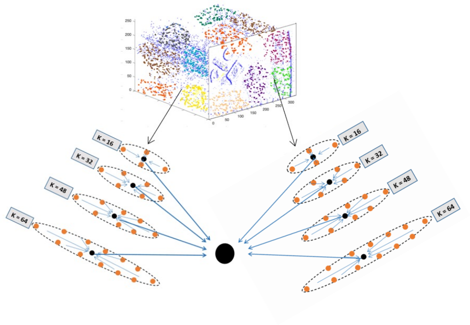

Figure 4. Collaborative Contextual Mixing depicting CCM intra-set and Inter-Set feature mixing.

$$u_{k,i} = \sum_{j \in M_{k,i}} softmax(s_j) * g_3(x_j) \qquad (5)$$

where the softmax function normalizes the spatial dimension. The symbol '$*$' indicates element-wise multiplication. Let $u_{k,i}$ denote the new feature vector obtained after aggregating k adjacent nodes. As the KNN is applied 4 times it will produce output feature vectors for each level, namely $u_{1,i}$, $u_{2,i}$, $u_{3,i}$, and $u_{4,i}$ and are subsequently aggregated using weighted sum as follows to obtain the final output feature vector u_i:

$$u_i = w_1 u_{1,i} + w_2 u_{2,i} + w_3 u_{3,i} + w_4 u_{4,i} \qquad (6)$$

w_1, w_2, w_3, and w_4 represent the weights assigned to each output feature vector. Followed by intra-set mixing, to mix the information between the sets the inter-set mixing method is applied. This can be understood as the inverse of the kNN, and the inverse index mapping $M^{-1}_{k,i}$ is defined (see Eq.8, which finds the set of indices j that includes event e_i. In this manner event features from the neighboring event sets are mixed [6].

3.2.3 Transition down block

The first step in the transition down block of the proposed method is to perform sampling of graph nodes using the farthest point sampling algorithm. The resulting sample nodes, denoted as G_s, are a subset of the original graph nodes G_o, i.e., $G_s \subset G_o$. The downsampled nodes are then used to compute their neighbors in the original graph using the kNN algorithm at four different levels k, which produces index maps denoted as M_k. By applying Eq.4 and Eq.5 with the calculated index mapping, the features of the original graph nodes are passed to the sample graph nodes.

For an original graph G with E_i nodes denoted as $G(E_i)$, the kNN algorithm is used to downsample it to $G(e_i)$. Mathematically,

$$M^s_{k,i} = KNN(E_i, k, e_i) \tag{7}$$

The kNN algorithm acts as a reduction factor in the transition down block, reducing the cardinality of the 3D graph and enabling the convolution of graph nodes. Specifically, if the original graph G has N nodes and a requested reduction factor of 4, the transition down module produces a new graph with $N/4$ nodes.

3.2.4 Transition up block

The transition-up block samples graph nodes from the transition-down and original graph node sets, without using the kNN function due to asymmetric neighbors. It utilizes the index mapping computed in the transition-down block to apply inverse mapping and upsample the nodes as:

$$M^{-1}_{k,i} = \{j | i \in M_j\} \tag{8}$$

The resulting mapping is used to calculate the original sample using Eq.4 and Eq.5, as shown in the Fig. 2. The transition-up module maps features from reduced graph dataset, G'(p2), to its superset, G'(p1) (where G'(p1) $\supset$ G'(p2)), without requiring an additional kNN search. Concatenating interpolated features of G'(p1) with the corresponding encoder stage's features via a skip connection enhances the features learned at the same level of the transition down block.

3.3. Proposed Network Architecture

The proposed GMNN architecture (Fig. 3) has four components: MLP blocks, transition down, transition up, and Mixer blocks, which form the encoder and decoder. The 3D event graph is passed through the encoder's MLP layer, followed by four downsampling levels that reduce node numbers by a factor of four each. The Mixer block uses the novel CCM method to spread features in parallel. The output of the encoder is then passed into the decoder, which begins with an MLP, followed by four upsampling levels with a factor of four each. Graph nodes are upsampled using the transition up block, and the Mixer block spreads features using the proposed CCM method in 3.2.4. The header block includes MLPs without a pooling layer for dense prediction tasks. The architecture adopts a deep pyramid-style structure to obtain global features by progressively downsampling nodes. In contrast to the conventional Graph Neural Network (GNN) approach, where graphs are constructed prior to network nonlinear operations and predictions, the Graph Mixture Neural Network (GMNN) employs a novel strategy of constructing subgraphs from each input graph and processing them in parallel within the Mixer layer and sampling modules. This approach facilitates the identification of spatiotemporal correlations between events and effectively captures the motion dynamics.

4. Experiments

4.1. Dataset

The ESD dataset [12] includes 17,186 annotated images and 177 labeled event streams, captured using a Davies346 sensor mounted at end of the robotic arm. It has variations in camera motion, arm speed, lighting conditions, and cluttered scenes. The dataset has instance-wise annotations for 15 object classes grouped into 6 categories. The training set (ESD-1) consists of 13,984 images of 10 known objects, while the testing set (ESD-2) consists of 3,202 images of 5 unknown objects that are not in ESD-1. Please refer to [12] for a detailed explanation of the experimental setup.

4.2. Training

In this study, in order to enable a fair comparison with state-of-the-art (SOTA) methods, we adopt an *effective* training scheme proposed by [2], that takes advantage of large amounts of training data while reducing computational requirements. The proposed scheme involves dividing the full training dataset into L subsets and exposing the neural network to only one of the subsets during each iteration, while the remaining subsets remain inactive. The network trains on a specific subset once every L iteration, resulting in complete training on the entire dataset after L epochs. This differs from conventional training methods where the entire dataset is used to update the neural network weights in every epoch. During the training process, the SGD optimizer is employed with a learning rate of 0.001 to minimize loss. In this study, each 3D event node is assigned to one of ten semantic categories, and the evaluation protocol suggested by Point Transformer is closely followed to ensure fairness. We use the SGD optimizer with a batch size of 4 during training and set the momentum and weight decay values to 0.9 and 0.0001 respectively. The weights w_1, w_2, w_3, and w_4 are empirically set to 0.10, 0.20, 0.30, and 0.40 respectively. The time interval is set to $T = 100ms$ and the maximum number of nodes to $N_{max} = 10000$, while the maximum spatial distances depend on the resolution of the event camera, therefore $X = 346$ and $Y = 260$.

4.3. Evaluation Metrics

Pixel accuracy and mIoU are used to evaluate the performance of panoptic segmentation. Pixel accuracy calculates the percentage of pixels in the image that are classified correctly. To adapt pixel accuracy to event-based vision data, the ratio for each object count of predicted events to ground truth events is calculated. Mean accuracy is then calculated across all objects. This approach provides a way to evaluate object detection models based on event data and accounts for the sparsity of events:

$$Acc(d, d') = \frac{1}{N} \sum_{1}^{N} \frac{d_i}{d'_i} \tag{9}$$

where d, d', and N represent the ground truth event set, the predicted event set, and the total number of events respectively.

The mIoU, also known as the Jaccard Index, handles better imbalanced binary and multi-class segmentation and is calculated across classes according to Eq.10:

$$mIoU = \frac{1}{C} \sum_{i}^{C} \frac{\sum_i^N \delta(d_{i,c}, 1)\delta(d_{i,c}, d'_{i,c})}{max(1, \delta(d_{i,c}, 1) + \delta(d'_{i,c}, 1)} \tag{10}$$

4.4. Quantitative Evaluation

In order to assess the efficacy of GMNN for panoptic segmentation, we present an evaluation of each sub-task of the dataset, which includes variations in the number of objects, lighting conditions, motion direction, camera speed, and object size across the entire dataset. Further, a similar evaluation is conducted on the unknown dataset to understand the model accuracy on unknown object segmentation.

The first experiment used a subset of the testing dataset with varying clutter levels of 2, 4, 6, 8, and 10 objects. More objects meant more occlusions and a more challenging scenario, evident in the segmentation accuracy results shown in experiment 1 of the table, 1. Accuracy scores for state-of-the-art models decreased from 82%-89% for 2 objects to 51%-70% for 10 objects, a reduction of 31%-19%. In contrast, the proposed GMNN model dropped only by 10% and outperformed the other methods for any number of scene objects.

Table 1. Segmentation accuracy of known objects in various conditions.

Exp 1: **varying clutter objects**, Bright light, 62cm height, Rotational motion, 0.15 m/s speed

Method	2 Obj	4 Obj	6 Obj	8 Obj	10 Obj
EV-SegNet [3]	82%	73%	67%	54%	51%
ESS [28]	86%	76%	68%	64%	60%
GTNN [2]	89%	86%	84%	77%	71%
GMNN (ours)	**97%**	**96%**	**91%**	**89%**	**87%**

Exp 2: 6 Objects, **varying lighting conditions**, 62cm height, Rotational Motion, 0.15 m/s speed.

Method	Bright Light	Low light
EV-SegNet [3]	76%	75%
ESS [28]	79%	78%
GTNN [2]	81%	79%
GMNN (ours)	**95%**	**94%**

Exp 3: 6 Objects, Bright Light, 62cm height, **Varying directions of motion**, 0.15 m/s speed.

Method	Linear	Rotational	Partial Rotational
EV-SegNet [3]	65%	73%	69%
ESS [28]	68%	78%	74%
GTNN [2]	75%	89%	78%
GMNN (ours)	**84%**	**93%**	**90%**

Exp 4: 6 Objects, Bright Light, 62cm height, Rotational motion, **Varying speed**.

Method	0.15 m/s	0.3 m/s	0.1 m/s
EV-SegNet [3]	69%	60%	56%
ESS [28]	72%	63%	59%
GTNN [2]	75%	71%	63%
GMNN (ours)	**93%**	**91%**	**87%**

Exp 5: 6 Objects, Bright Light, **Varying camera height**, Rotational motion, Varying speed.

Method	62 cm	82 cm
EV-SegNet [3]	76%	74%
ESS [28]	82%	75%
GTNN [2]	85%	83%
GMNN (ours)	**97%**	**93%**

State-of-the-art methods in comparison to GMNN show relatively poor segmentation accuracy in both bright lighting conditions, as shown in experiment 2 of table 1: 76%, 80% and 81% for EV-SegNet, ESS and GTNN respectively. These methods further drop the accuracy in low light conditions: 75%, 78%, and 79% for the EV-SegNet, ESS, and GTNN respectively. The proposed GMNN seems to be robust against varying lighting conditions as it achieves the highest accuracy in both lighting conditions i.e. 95% in bright and 94% in dark light.

The subsequent experiment was conducted on a subset of the testing dataset where the robotic arm movement direction was varied as linear, rotational, or partial rotational. In event-based vision sensors, the direction of the robotic arm movement plays a crucial role as perpendicular edges generate more informative event sets compared to parallel edges. The impact of the phenomenon is demonstrated in experiment 3 of table 1, where EV-SegNet, ESS, and GTNN models have the highest accuracy score for rotational motion 73%,78%, 89% respectively, decreasing in partial rotational to 69%,74%, 78% respectively and the lowest for the linear motion 65%,68%, 75% respectively. In contrast, the proposed GMNN achieves the highest average accuracy score of 93% in rotational motion, decreasing for partial rotational motion and linear motion to only 90% and 84% respectively.

The camera is placed at the end of the robotic arm thus the camera speed is an important factor while evaluating the robustness of the model. The next experiment was conducted on a subset of the training dataset where the speed of the end effector was varied. As can be seen in experiment 4 of table 1, state-of-the-art models have an accuracy of 69% to 75% for 0.15 m/s, which drops to 60% to 71% for 1m/s. Whereas the proposed GMNN model has the highest accuracy of 93% for 0.15m/s and it drops to 91% for 1m/s. The clear impact of the CCM mixing layer in high-speed conditions supports the recovery of the information at contours.

In order to understand the scale invariance of the model an experiment was conducted on a subset of the training dataset where the distance between the platform and the camera was varied to 62 cm and 82 cm. As per the results illustrated in experiment 5 of the Table. 1 there is a minimal impact of the camera and object distance on the accuracy of all the models, and GMNN maintains its superiority.

Table 2 compares the performance of four methods, EV-SegNet [3], ESS [28], GTNN [2], and GMNN, on the Known Object Dataset and the ESD-2 Unknown Object Dataset. The proposed GMNN method achieved the highest mIoU and accuracy, outperforming all other architectures, while EV-SegNet and ESS had the lowest performance, and GTNN achieved a 74.24% mIoU, which was still 3.5% lower than the proposed GMNN model.

However, the performance of all methods dropped when evaluated on the Unknown Object Dataset. The proposed GMNN achieved 89.91% accuracy, while graph-based methods dropped by 6% compared to EV-SegNet and ESS. These results justify the use of graph structure for asynchronous events for segmentation challenges as it learns the temporal relationships in a better fashion.

Table 2. Quantitative comparison of GMNN against other Asynchronous event fusion methods on the whole ESD dataset.

Methods	Known Obj		Unknown Obj	
	mIoU %	Acc %	mIoU %	Acc %
EV-SegNet [3]	7.73	76.98	5.29	53.31
ESS [28]	8.92	81.59	7.01	67.29
GTNN [2]	74.24	87.53	58.70	81.30
GMNN (ours)	**78.32**	**96.91**	**66.05**	**89.91**

4.5. Model Size

Table 3 compares the amount of parameters of GMNN against the other three state-of-the-art methods for panoptic segmentation: EV-SegNet [3], ESS [28], and GTNN [2]. GMNN utilizes the least number of parameters at 3.9 million, while the other methods require significantly more, ranging from 5.3 to 22 million. This suggests that GMNN may be more efficient and scalable in terms of model size and training time.

Table 3. Comparison of model size of GMNN against other Asynchronous event fusion methods

Methods	**Parameters**
EV-SegNet [3]	22M
ESS [28]	17M
GTNN [2]	5.3M
GMNN (ours)	**3.9M**

4.6. Computational Time analysis

Table 4 displays the computational time taken by the GMNN and GTNN models, implemented on a Dell desktop and a Google Colab's NVIDIA Tesla K80 GPU. The analysis of 40 event graphs, each lasting 10ms, was conducted in two modes of operation, namely sequential and batch mode. The sequential model in PyTorch processes event graphs successively, while the batch mode processes event graphs as a single batch. The results demonstrate that GMNN outperforms GTNN in terms of computational time in both modes of operation. It should be noted that the graph is constructed using events within 100ms, and its size varies from 1 to a maximum of 10000 events, depending on the number of events triggered within the temporal window. Thus, calculating the standard deviation is essential in understanding the computational time performance. Overall, the proposed approach achieves a significant speed-up of up to one order of magnitude in both batch and sequential modes, which is necessary to handle batches of events concurrently while preserving the high temporal resolution of the sensor.

Table 4. Comparison of Sequential and Batch Modes

Model	**Sequential-mode**	**Batch-mode**
	$\mu \pm \sigma$ (sec)	μ (sec)
GTNN	$9.63 \times 10^{-2} \pm 1.93 \times 10^{-4}$	13.52×10^{-4}
GMNN	$\mathbf{4.06 \times 10^{-3} \pm 2.05 \times 10^{-4}}$	$\mathbf{10.27 \times 10^{-5}}$

5. Ablation Study

We conduct an ablation study about the proposed CCM in semantic segmentation on the ESD dataset to understand the best suitable combinations of nearest neighbors and a number of parallel features mixing.

5.1. Parallel Mixing within CCM

Table 5 investigates the effect of varying the number of K-Nearest Neighbours (kNN) in each layer of the CCM on accuracy in deep learning models. Results indicate that an increase in the number of kNN layers generally improves panoptic segmentation accuracy for up to 4 layers, however increasing the number of layers further leads to a decrease. This emphasizes the importance of optimizing the number of layers and kNNs in the CCM for maximum accuracy. Note that the PointMixer [6]method is equivalent to using only one kNN layer.

Table 5. Parametric comparison of GMNN against other Asynchronous event fusion methods.

	kNN in Each Layer of CCM							
	L1	**L2**	**L3**	**L4**	**L5**	**L6**	**L7**	**ACC%**
1	16	-	-	-	-	-	-	81.23
2	16	32	-	-	-	-	-	89.04
3	16	32	48	-	-	-	-	92.50
4	16	32	48	64	-	-	-	**96.91**
5	16	32	48	64	80	-	-	96.05
6	16	32	48	64	80	96	-	95.37
7	16	32	48	64	80	96	112	93.75

5.2. Impact of varying the kNN size

Table 6 investigates the effect of different sets of nearest neighbors in the CCM layer on the accuracy. Oversmoothing can occur in CCM when too many features are stacked together. The study found that increasing the set of k initially improves accuracy, but Set 4 and Set 5 resulted in decreased accuracy due to over-smoothing.

Table 6. Parametric comparison of GMNN against other Asynchronous event fusion methods.

Set of k	**Layer 1**	**Layer 2**	**Layer 3**	**Layer 4**	**Layer 5**
Set 1	3	3	9	12	89.23%
Set 2	8	16	24	32	93.04%
Set 3	16	32	48	64	**96.91**%
Set 4	25	50	75	100	95.19%
Set 5	40	80	160	240	92.50%

5.3. Misclassified Boundary

Table 7 compares four panoptic segmentation methods for known and unknown object subsets using TP, FP, TN, and FN evaluation metrics. The percentage of misclassified events, representing the percentage of FP events that overlap with object boundaries, was measured using 84000 events for known objects and 32000 events for unknown objects. Segmenting the boundaries of an object is significant for robotic grasping because it allows the robot to precisely locate and segment the object, enabling planning and executing more accurate grasping motions, improving efficiency and reducing the risk of mishandling. The GMNN method had the lowest percentage of the misclassified events on the boundaries of 4% for known and 10% for unknown object subsets, while other methods ranged from 10% to 12% for known and 20% to 35% for unknown objects. These findings suggest that the GMNN method is better at identifying object edges, making it a promising option for robotic grasping in challenging conditions.

Table 7. Analysis of event overlap.

Method	TP	FP	TN	FN	Overlapped Events (% of FP)
			Known Object		
EV-SegNet [3]	20160	10080	41160	12600	12%
ESS [28]	25200	9240	42840	6720	11%
GTNN [2]	36960	8400	35280	3360	10%
GMNN (ours)	42000	3360	37800	840	**4%**
			Unknown Object		
EV-SegNet [3]	4902	11098	14184	1816	35%
ESS [28]	3778	12243	15128	872	38%
GTNN [2]	9444	6556	13580	2420	20%
GMNN (ours)	12806	3194	13026	2974	**10**%

6. Conclusion

The study proposes an approach to panoptic segmentation using a dynamic vision sensor and integrating the novel Collaborative Contextual Mixing (CCM) technique with the U-Net framework. The architecture combines neighboring events at multiple levels and produces a parallel feature learning representation. The encoder performs downsampling operations while the decoder executes upsampling operations on events, resulting in an effective panoptic segmentation model for robotic grasping.

Our proposed model performs exceptionally well on the ESD dataset under diverse conditions and achieves state-of-the-art results in terms of mIoU and pixel accuracy, demonstrating the robustness of the introduced CCM approach against challenges like occlusions, low lighting, small objects, high speed, and linear motion. Additionally, our method utilizes GNNs and mixer techniques, resulting in shorter prediction time than existing state-of-the-art methods.

Future research could explore the proposed approach's generalization capability in real-world scenarios with diverse robots, sensors, and environments, and incorporating other sensors like depth sensors or thermal cameras that could improve the model's low-light performance.

Acknowledgements

This academic publication was funded by Kingston University and acknowledges the significant contributions of researchers from Khalifa University. In addition, the authors would like to extend their appreciation to Ipsotek, an Eviden Company, for their generous support and valuable contributions to the research project.

References

[1] Yusra Alkendi, Rana Azzam, Abdulla Ayyad, Sajid Javed, Lakmal Seneviratne, and Yahya Zweiri. Neuromorphic Camera Denoising Using Graph Neural Network-Driven Transformers. *IEEE Transactions on Neural Networks and Learning Systems*, 2022. 1, 2

[2] Yusra Alkendi, Rana Azzam, Sajid Javed, Lakmal Seneviratne, and Yahya Zweiri. Neuromorphic Vision-based Motion Segmentation with Graph Transformer Neural Network. Technical report. 1, 2, 5, 6, 7, 8

[3] Iñigo Alonso and Ana C Murillo. EV-SegNet: Semantic Segmentation for Event-based Cameras. *IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), Long Beach, CA, USA*, pages 1624–1633, 2019. 2, 6, 7, 8

[4] Chaoqi Chen, Yushuang Wu, Qiyuan Dai, Hong-Yu Zhou, Mutian Xu, Sibei Yang, Xiaoguang Han, and Yizhou Yu. A Survey on Graph Neural Networks and Graph Transformers in Computer Vision: A Task-Oriented Perspective. 9 2022. 2

[5] Liang Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L. Yuille. DeepLab: Semantic Image Segmentation with Deep Convolutional Nets, Atrous Convolution, and Fully Connected CRFs. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(4):834–848, 2018. 2

[6] Jaesung Choe, Chunghyun Park, Francois Rameau, Jaesik Park, and In So Kweon. PointMixer: MLP-Mixer for Point Cloud Understanding. *arXiv preprint arXiv:2111.11187*, 11 2022. 2, 3, 5, 8

[7] Venkatesa Prabu Dinakaran, Meenakshi Priya Balasubramaniyan, Quynh Hoang Le, Ali Jawad Alrubaie, Ameer Al-khaykan, Suresh Muthusamy, Hitesh Panchal, Mustafa Musa Jaber, Anil Kumar Dixit, and Chander Prakash. A novel multi objective constraints based industrial gripper design with optimized stiffness for object grasping. *Robotics and Autonomous Systems*, 160, 2 2023. 1

[8] Hongyang Gao and Shuiwang Ji. Graph U-Nets. *arXiv preprint arXiv:1905.05178*, 5 2019. 2

[9] Yunjun Gao, Baihua Zheng, Gencai Chen, Wang Chien Lee, Ken C.K. Lee, and Qing Li. Visible reverse k-nearest neighbor query processing in spatial databases. *IEEE Transactions on Knowledge and Data Engineering*, 21(9):1314–1327, 9 2009. 3

[10] Sourav Garg, Niko Sünderhauf, Feras Dayoub, Douglas Morrison, Akansel Cosgun, Gustavo Carneiro, Qi Wu, Tat-Jun Chin, Ian Reid, Stephen Gould, Peter Corke, and Michael Milford. Semantics for Robotic Mapping, Perception and Interaction: A Survey. *Foundations and Trends® in Robotics*, 8(1–2):1–224, 2020. 1

[11] Fuqiang Gu, Weicong Sng, Tasbolat Taunyazov, and Harold Soh. TactileSGNet: A Spiking Graph Neural Network for Event-based Tactile Object Recognition. 7 2020. 2

[12] Xiaoqian Huang, Kachole Sanket, Abdulla Ayyad, Fariborz Baghaei Naeini, Dimitrios Makris, and Yahya Zweiri. A Neuromorphic Dataset for Object Segmentation in Indoor Cluttered Environment. *arXiv preprint arXiv:2302.06301*, 2 2023. 1, 5

[13] Sanket Kachole, Xiaoqian Huang, Fariborz Baghaei Naeini, Rajkumar Muthusamy, Dimitrios Makris, and Yahya Zweiri. Bimodal SegNet: Instance Segmentation Fusing Events and RGB Frames for Robotic Grasping. *arXiv preprint arXiv:2303.11228*, 2023. 2

[14] Huayao Liu, Jiaming Zhang, Kailun Yang, Xinxin Hu, and Rainer Stiefelhagen. CMX: Cross-Modal Fusion for RGB-X Semantic Segmentation with Transformers. *arXiv preprint arXiv:2203.04838*, 3 2023. 2

[15] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. *Computer Vision and Pattern Recognition*, pages 431–440, 2015. 2

[16] Jinkai Lv, Yuyong Hu, Quanshui Fu, Zhiwang Zhang, Yuqiang Hu, Lin Lv, Guoqing Yang, Jinpeng Li, and Yi Zhao. CM-MLP: Cascade Multi-scale MLP with Axial Context Relation Encoder for Edge Segmentation of Medical Image. In *Proceedings - 2022 IEEE International Conference on Bioinformatics and Biomedicine, BIBM 2022*, pages 1100–1107. Institute of Electrical and Electronics Engineers Inc., 2022. 3

[17] Rodrigo Marcuzzi, Lucas Nunes, Louis Wiesmann, Jens Behley, and Cyrill Stachniss. Mask-Based Panoptic LiDAR Segmentation for Autonomous Driving. *IEEE Robotics and Automation Letters*, 8(2):1141–1148, 1 2023. 1

[18] Rohit Mohan and Abhinav Valada. Amodal Panoptic Segmentation. *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 20991–21000, 10 2022. 1

[19] Giuseppe Montano, Marek Rucinski, Elie Allouis, Olivier Notebaert, and David Jameux. Network latency analysis of a SpaceWire-based control system for space robotic arm: SpaceWire missions and applications, short paper. In *Proceedings of the 2016 7th International SpaceWire Conference, SpaceWire 2016*. Institute of Electrical and Electronics Engineers Inc., 12 2016. 1

[20] Senthilkumaran N and Vaithegi S. Image Segmentation By Using Thresholding Techniques For Medical Images. *Computer Science & Engineering: An International Journal*, 6(1):1–13, 2 2016. 2

[21] Fariborz Baghaei Naeini, Sanket Kachole, Dimitrios Makris, and Yahya Zweiri. Event Augmentation for Contact Force Measurements. *IEEE Access*, 10:123651–123660, 2022. 2

[22] Fariborz Baghaei Naeini, Dimitrios Makris, Dongming Gan, and Yahya Zweiri. Dynamic-vision-based force measurements using convolutional recurrent neural networks. *Sensors (Switzerland)*, 20(16):1–15, 8 2020. 2

[23] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 9351, pages 234–241. Springer Verlag, 2015. 2

[24] Alberto Sabater, Luis Montesano, and Ana C. Murillo. Event Transformer. A sparse-aware solution for efficient event data processing. 4 2022. 2

[25] Kachole sanket, Mahakal Manish, and Bhagwatkar Anurag. 3 Dimensional Welding SPM/Path Tracker. *International Journal Of Design And Manufacturing Technology*, 7(3), 12 2016. 1

[26] Simon Schaefer, Daniel Gehrig, and Davide Scaramuzza. AEGNN: Asynchronous Event-based Graph Neural Networks. *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 12361–12371, 2022. 2

[27] Stefan Schliebs and Nikola Kasabov. Evolving spiking neural network-a survey. *Evolving Systems*, 4(2):87–98, 6 2013. 2

[28] Zhaoning Sun, Nico Messikommer, Daniel Gehrig, and Davide Scaramuzza. ESS: Learning Event-based Semantic Segmentation from Still Images. *arXiv preprint arXiv:2203.10016*, 2022. 2, 6, 7, 8

[29] S. Thilagamani and N. Shanthi. Object recognition based on image segmentation and clustering. *Journal of Computer Science*, 7(11):1741–1748, 2011. 2

[30] Hai Wang, Yanyan Chen, Yingfeng Cai, Long Chen, Yicheng Li, Miguel Angel Sotelo, and Zhixiong Li. SFNet-N: An Improved SFNet Algorithm for Semantic Segmentation of Low-Light Autonomous Driving Road Scenes. *IEEE Transactions on Intelligent Transportation Systems*, 23(11):21405–21417, 11 2022. 1

[31] Weihao Yu, Mi Luo, Pan Zhou, Chenyang Si, Yichen Zhou, Xinchao Wang, Jiashi Feng, and Shuicheng Yan. MetaFormer is Actually What You Need for Vision. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 2022-June, pages 10809–10819. IEEE Computer Society, 2022. 2

[32] Hengshuang Zhao, Li Jiang, Jiaya Jia, Philip Torr, and Vladlen Koltun. Point Transformer. *arXiv preprint arXiv:2012.09164*, 12 2020. 2