

# ULRR

## **BEVSeg2GTA: Joint vehicle segmentation and graph neural networks for ego vehicle trajectory prediction in bird's-eye-view**

|               |   |
|---------------|---|
| Item Type     | Article   |
| Authors       | Sharma, Sushil;Das, Arindam;Sistu, Ganesh;Halton, Mark;Eising, Ciarán   |
| Citation      | IEEE Access, 2024, 12, pp. 132159-132174  |
| Publisher     | Institute of Electrical and Electronics Engineers   |
| Download date | 2026-04-22 07:32:48   |
| Item License  | <a href="https://creativecommons.org/licenses/by-nc-sa/4.0/">https://creativecommons.org/licenses/by-nc-sa/4.0/</a>           |
| Link to Item  | <a href="https://doi.org/10.34961/researchrepository-ul.27629550">https://doi.org/10.34961/researchrepository-ul.27629550</a> |

## RESEARCH ARTICLE

# BEVSeg2GTA: Joint Vehicle Segmentation and Graph Neural Networks for Ego Vehicle Trajectory Prediction in Bird's-Eye-View

SUSHIL SHARMA<sup>1,2,3</sup>, ARINDAM DAS<sup>1,3</sup>, GANESH SISTU<sup>1,3</sup>,  
MARK HALTON<sup>1,3</sup>, (Member, IEEE),  
AND CIARÁN EISING<sup>1,2,3</sup>, (Senior Member, IEEE)

<sup>1</sup>Department Electronic and Computer Engineering, University of Limerick, Limerick, V94 T9PX Ireland

<sup>2</sup>SFI CRT Foundations in Data Science, University of Limerick, Limerick, V94 T9PX Ireland

<sup>3</sup>Data-Driven Computer Engineering (D<sup>2</sup>ICE) Research Centre, University of Limerick, Limerick, V94 T9PX Ireland

Corresponding author: Sushil Sharma (sushil.sharma@ul.ie)

This work was supported by the Science Foundation Ireland under Grant 18/CRT/6049.

**ABSTRACT** Predicting the trajectory of the ego vehicle is a critical task for autonomous vehicles. Even though traffic regulations have defined boundaries, various behaviors of the agents in real-life situations introduce complexities that are hard to capture comprehensively. This has led to a rising curiosity in ego vehicle trajectory prediction based on learning techniques. In this paper, we introduce *BEVSeg2GTA* (Bird's-Eye-View Joint Vehicle Segmentation and Graph Neural Network Trajectory Prediction), a novel approach that aims to forecast trajectories by treating perception and trajectory prediction as interconnected elements of a single system. By integrating these tasks, we demonstrate the possibility of improving perception accuracy and trajectory prediction error. Initially, an encoder-decoder transformer-based deep network is employed to convert the multi-view camera images to a Bird's-Eye-View representation followed by semantic segmentation of crucial agents, including the ego vehicle, other vehicles, and pedestrians within the scene. Integrating state-of-the-art backbone (such as EfficientNet) facilitates the extraction of strong features, which are used to construct a graph wherein a node represents each object within the scene. Subsequently, the connections of these nodes are established by a  $k$ -Nearest Neighbors algorithm based on the distance metric. Further, the node and image features are fed into a Graph Neural Network to learn the complex relationships between agents in a spatial context. Finally, the Graph Neural Network learned features are passed to a Spatio-Temporal Probabilistic Network to predict the ego vehicle's future trajectory accurately. The proposed framework, *BEVSeg2GTA*, has been extensively evaluated on nuScenes datasets. The results demonstrate that the proposed method improves the state-of-the-art performance.

**INDEX TERMS** Multi-view camera, encoder-decoder transformer, Bird's-Eye-View, semantic segmentation, graph neural network, spatio-temporal probabilistic network, trajectory prediction.

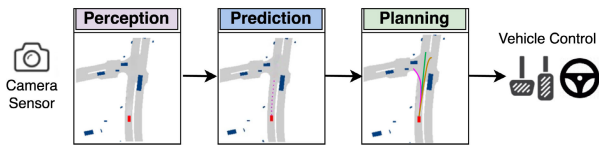
## I. INTRODUCTION

The development of autonomous driving technologies emphasizes the importance of safety [1], [2], which relies on continuous coordination between perception, prediction, planning, and control systems, as depicted in Figure 1. Predicting the trajectory of the ego vehicle is an essential

The associate editor coordinating the review of this manuscript and approving it for publication was Diego Oliva<sup>1</sup>.

element in the development of autonomous driving systems [3], [4], [5]. It allows vehicles to navigate through complex traffic situations [6], [7]. This task is particularly challenging due to the unpredictable nature of human behavior and the dynamic interactions between agents (i.e., vehicles and pedestrians).

Accurate ego trajectory prediction is used for understanding the complex spatial and temporal relationships in dynamic scenes [8], [9], [10], [11]. Thus, this can potentially enhance



**FIGURE 1.** Modern self-driving systems include standard components in their setup, outlining various tasks.

autonomous driving systems' safety, efficiency, and overall performance. For example, ego trajectory prediction enables autonomous vehicles to optimize their driving behavior, leading to smoother lane changes and more efficient vehicle merges [12]. This ultimately contributes to an overall improvement in traffic flow.

In this study, we introduce a novel method, namely BEVSeg2GTA, aimed at improving ego vehicle trajectory prediction by utilizing an encoder-decoder transformer [13] and Graph Neural Network (GNN) models [14]. This work uses GNNs to learn the complex relationships between vehicles and their interactions within the traffic environment. These networks are capable of capturing the dependencies between vehicles, aiding in the prediction of their future trajectories based on the current state of the network, with the use of a Spatio-Temporal Probabilistic Network (STPN) [15].

Multi-camera views provide a 360° perspective of the surrounding environment, capturing details that a single camera might miss. Our strategy involves employing a network trained on multi-camera views captured by the host vehicle, which converts these views into Bird's-Eye-View (BEV) representations of the surrounding environment, providing a comprehensive, simplified view of the environment. These representations undergo deep learning-based semantic segmentation processes to identify objects within the scene, including pedestrians, other vehicles, and ego vehicles, as detailed in [13]. The results of this segmentation are then integrated across different camera perspectives to create a BEV representation of the surrounding vehicles.

Building on the segmented data, our proposal uses segmentation masks to identify specific regions within images. These regions are then utilized for feature extraction via a ResNet-18 [16] network. The extracted features are subsequently used in a  $k$ -Nearest Neighbor ( $k$ NN) [17] algorithm to create edges that connect with nodes.  $k$  refers to a positive integer that defines the closest training examples in the data set. Consequently, a graph is constructed, with bounding box features as nodes and edges derived from the  $k$ NN approach. To capture spatial patterns within this graph representation, we employ a GNN model, which learns spatio-temporal features for trajectory prediction. Finally, the host vehicle utilizes an STPN, as described by [15]. The STPN learns the spatial patterns of vehicle motion from historical trajectory data. The predicted trajectories are then projected back to the ego vehicle's BEV perspective, providing an understanding

of the surrounding traffic dynamics. Figure 2 gives a conceptual overview of our approach.

The main contribution of our paper is to propose an architecture that brings together components of Camera to BEV Transformation, BEV Segmentation, GNNs, and STPNs to provide a network that jointly estimates the BEV segmentation and the ego vehicle trajectory:

- Our proposal architecture (**BEVSeg2GTA**) offers an approach to accomplish vehicle segmentation and ego vehicle trajectory prediction tasks jointly, thereby achieving state-of-the-art results in both segmentation and trajectory prediction compared on the nuScenes dataset [18]
- Our proposal integrates GNNs into the BEV architecture to capture the spatial relationships of the host vehicle and nearby objects, where the weights of the connections are inverses of the distances.
- STPN is then integrated for trajectory estimation, so our network proposal considers both spatial and temporal aspects of the problem.

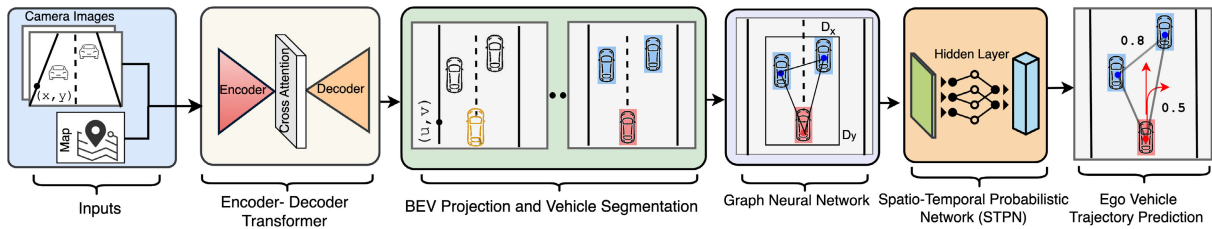
Preliminary results of this work have been published in [19]. In this paper, we have extended and improved upon our previous work as follows:

- To learn the complex relations of different road agents, a GNN is integrated into this proposal.
- Theoretical explanations of each component are added with significantly greater details.
- We have done a systematic ablation study to evaluate the effect of proposed individual components, parameters, losses, and different evaluation settings.
- We added a detailed discussion on the limitations and failure scenarios of the proposed method.

The structure of the paper is as follows: Section II provides an overview of related works in the field. In Section III, we delve into a detailed description of the proposed network. Section IV outlines the experimental setup used to obtain the results presented in this paper, including an ablation study in Section IV-E. In Section V, we present the experimental results and analyze their implications. In Section V-C, we present the failure scenario in predicting the ego vehicle's trajectory. Lastly, Section VI concludes the paper by summarizing the findings and outlining directions for future research.

## II. LITERATURE REVIEW

This section reviews related work on trajectory prediction in autonomous vehicles. We outline common problem formulations and solutions, focusing on model-driven and data-driven approaches. These methods address key challenges in predicting vehicle trajectories, offering valuable insights into current techniques used in the field. Table 1 illustrates our choice of the nuScenes dataset. We selected this dataset because it offers six camera views, providing a comprehensive top-down view that facilitates clear and accurate BEV representation. This multi-view perspective



**FIGURE 2.** The BEVSeg2GTA framework aims to enhance ego vehicle trajectory prediction by integrating an encoder-decoder transformer and a GNN. The framework begins with a projection module, which transforms the input multi-camera views and map information into a BEV perspective using an encoder-decoder transformer. The segmentation output from this module is then fed into the GNN, which is utilized to construct a graph representing spatial information. This graph is subsequently employed as input into an STPN, which produces predictions of the ego vehicle's trajectory.

allows for a detailed capture of the environmental layout from a top-down perspective

### A. BIRD'S-EYE-VIEW REPRESENTATION

BEV-based perception involves converting perspective images to a top-view representation and performing various tasks such as 3D detection [20], [21], [22], map segmentation [23], [24], tracking, and motion planning [25], [26]. Due to its inherent advantages in 3D spatial representation [27], [28], multi-modal fusion [29], [30], decision-making, and planning, the subject of BEV perception has aroused great interest from researchers in academia and industry. Recent advances in object detection have led to the development of CNN-based 3D object detection models [31], [32], [33], [34] capable of generating 3D bounding boxes (BBOX) from monocular images. It is observed that these models have shown promising results in the automotive field. One such approach is to use the DETR [35] architecture as the basis for a 3D detection model using only surround-view images, which has led to the development of several camera 3D detection models leveraging Transformer-based models. These models extract relevant camera features from multiple views using object queries employing spatial cross-attention (e.g., the Cross-view Transformer (CVT) [13]). BEVFormer [36] further improves detection and segmentation performance by employing BEV spatial queries and temporal attention to previous frames. The Lift, Splat, Shoot (LSS) [37] method integrates a surround-view camera into BEV representation to improve detection performance, while CVT [13] and PowerBEV [38] integrate feature information to improve BEV Visibility, referring to the quality and detail of the top-down environmental view. Overall, the field of BEV is rapidly evolving, and new approaches are being developed to overcome the limitations of existing methods. Along this line of work, we improve the BEV Visibility using map information, enhancing accuracy and robustness and helping to predict future trajectories.

### B. MAP SEGMENTATION IN BIRD'S-EYE-VIEW

BEV-based semantic map segmentation can be categorized into two types: single-view and multi-view segmentation. Single-view segmentation [23], [39] faces the challenge

of creating the entire BEV semantic map from just one image taken by the front-view camera of the ego vehicle. To address this, earlier methods used synthetic datasets like CARLA [40] or CARSIM [41], which offer high-perspective views. Nowadays, with the availability of multi-camera view datasets like nuScenes [18] and Waymo [42], recent approaches [13], [38] use the camera intrinsic and extrinsic parameters to convert multiple views into BEV semantic maps. Map segmentation techniques fall into two main categories: transformer-based segmentation [13], [19], [36], [38], [43] and deep-CNN-based segmentation [36], [37], [44], [45]. CVT [13] uses camera-geometry aware transformers and positional embeddings to achieve map view segmentation of multi-view images. However, it mainly focuses on local patch information rather than the overall relationship between regions. Note that, CVT is proficient at analyzing specific local features, it may not effectively capture the holistic relationships and structures present in the overall region being analyzed as detailed in this paper [13]. The Collaborative Bird's Eye View Transformer (CoBEVT) [43] uses sparse attention to capture sparse global information by considering only a subset of global features instead of all relationships. LSS [37] introduces a CNN-based deep semantic segmentation network by considering the feature maps from each view and merging them into a single intermediate BEV representation.

### C. GRAPH NEURAL NETWORK IN BIRD'S-EYE-VIEW

Graph-based methods are gaining popularity for various tasks in BEV scenarios, including object detection [52], [53], semantic segmentation [54], [55], and scene understanding [56]. BEV provides a simplified view of the environment, facilitating the acquisition of contextual information [4]. The GNN model [57] utilizes spatial relationships between objects, represented as nodes in a graph, to capture complex interactions and dependencies. Additionally, GNNs detect and track objects such as vehicles, pedestrians, and cyclists in BEV representations [58], [59]. Our research employs GNNs to establish spatial relationships between agents and ego vehicles within the scene. Analyzing spatial relationships closer to the ego vehicle reveals connections between agents and indicates the strength of their relationship with nearby

**TABLE 1.** Dataset for autonomous vehicles utilized in trajectory prediction.

| Datasets                | Years | Agents   |             |         | Sensor |       |       | BEV | HD Map            | Others |
|-------------------------|-------|----------|-------------|---------|--------|-------|-------|-----|-------------------|--------|
|                         |       | Vehicles | Pedestrians | Cyclist | Camera | Lidar | Radar |     |                   |        |
| nuScenes [18]           | 2020  | ✓        | ✓           | ✓       | ✓      | ✓     | ✓     | ✓   | Multi-View Camera |        |
| CARLA [40]              | 2017  | ✓        | ✓           | ✓       | ✓      | ✓     | ×     | ✓   | -                 |        |
| Waymo Open Dataset [46] | 2020  | ✓        | ✓           | ✓       | ✓      | ✓     | ×     | ✓   | -                 |        |
| KITTI [47]              | 2013  | ✓        | ✓           | ✓       | ✓      | ✓     | ✓     | ×   | -                 |        |
| Argoverse [48]          | 2019  | ✓        | ×           | ×       | ✓      | ✓     | ✓     | ✓   | Stereo Cameras    |        |
| Lyft Level 5 [49]       | 2020  | ✓        | ✓           | ✓       | ✓      | ×     | ×     | ✓   | -                 |        |
| Interaction [50]        | 2019  | ✓        | ✓           | ×       | ✓      | ×     | ×     | ✓   | -                 |        |
| ApolloScape [51]        | 2018  | ✓        | ✓           | ✓       | ✓      | ✓     | ×     | ×   | -                 |        |

objects. This additional information enables more accurate trajectory predictions.

### D. BIRD'S-EYE-VIEW BASED TRAJECTORY PREDICTION

In the field of predicting the trajectory of autonomous vehicles, researchers commonly analyze data from sensors such as Lidar, Radar, and Cameras [60], [61], [62]. By integrating data from these sensors, they can accurately predict vehicle trajectories. The goal is to guess where the autonomous vehicles and other objects will be in the future. This is important for making autonomous driving systems safe and efficient. Using a BEV gives a complete view of the environment, including details on all objects like vehicles and pedestrians. This approach is supported by previous research [13], [19], [38], which has shown the benefits of using a bird's-eye view for understanding the surrounding traffic dynamics and optimizing the trajectory of the ego vehicle. Recent studies have explored trajectory prediction [63], [64] [65], [66], [67] task, primarily focusing on raster maps to forecast future trajectories. Transitioning from a perspective to a BEV for trajectory prediction presents unique challenges due to unavailable segmentation details. It's noteworthy that much of the existing work relies on sensor fusion [68], [69] and single monocular camera [70] incorporating information from additional sensors like Lidar and Radar. However, our method concentrates solely on camera-based information, significantly reducing computation time. In this study, we propose a solution to address these challenges. Our approach involves leveraging BEV representation in conjunction with surround-view images. Front-view perspective images offer detailed object information, which can be lost in a BEV, affecting trajectory predictions based on a spatial-temporal probabilistic network. We utilize surround-view cameras (**6-views** cameras) to capture complete scene information. We calculate vector information through graph embedding to aid in ego vehicle trajectory prediction. To achieve this, we employ a temporal probabilistic network to compute potential trajectories, selecting the most probable outcome after that. This outcome is then projected back onto the scene, completing the trajectory prediction process. The key difference in our proposed method is that graph neural networks (GNNs) are used to generate embeddings representing the environment around the ego vehicle. Finally, these embeddings are used

as input to a spatio-temporal probabilistic layer to predict the future trajectory of the ego vehicle.

### III. PROPOSED METHODOLOGY

Our proposal BEVSeg2GTA method, as described in Figure 3, utilizes multiple cameras to create a view of the environment around the ego vehicle, incorporating vehicle segmentation and the prediction of the ego vehicle's trajectory. This method extends the transformer technique for generating the BEV and then uses GNNs to establish spatial relationships between objects. This information is utilized to calculate probabilistic trajectories using an STPN. We provide both a summary of the main elements of the prior works and a focused description of how the individual methods are integrated into our proposed network architecture.

#### A. MULTI-PERSPECTIVE CAMERA INPUTS

The dataset employed in this study is the nuScenes dataset [18]. It comprises six cameras placed around the vehicle, providing a 360° field of view. Each camera within has extrinsic  $(R, t)$  and intrinsic  $M$  calibration parameters, which are provided at each timestamp, with the intrinsic parameters remaining constant over time. It should be noted that while the nuScenes dataset contains other perception sensors, such as radar and lidar, they are not used in this work.

#### B. CAMERA TO BEV TRANSFORMATION

In this section, we provide detailed explanations of all the components used in our Camera to BEV Transformation (CBT) block (refer to Figure 3). This block is designed to obtain a BEV representation of the scene by integrating features from each monocular camera and transforming them into the BEV space.

##### 1) POSITIONAL EMBEDDING

Later in Section III-B3, we will describe how Cosine Similarity is used in Cross-view Attention. Here, we improve the cosine similarity measure by including positional embeddings. This helps the model better understand both the geometric and visual aspects of the input images. For every point in the image, we create a direction vector that acts as a reference in the coordinate system. Then, we utilize a Multi-Layer Perceptron to transform these direction vectors



Initially, the image encoder creates a detailed feature representation  $\{\phi\}$  for each image. These features are then combined into a shared map-view representation through a cross-view cross-attention mechanism. This mechanism uses positional embeddings  $\{\delta\}$  to capture the scene's geometry for precise spatial alignment and the sequence of camera views for better temporal understanding and context integration. All camera-aware positional embeddings are combined into a single key vector  $\delta = [\delta_1, \delta_2 \dots \delta_6]$  and image features are merged into a value vector  $\phi = [\phi_1, \phi_2 \dots \phi_6]$ . These vectors are then compared to create attention keys, and a softmax-cross attention is applied to refine the map-view representation [71]. This process provides a BEV frame that includes information about the ego vehicle, other agents, and their positions in the scene.

### C. BIRD'S-EYE-VIEW SEGMENTATION & PROJECTION

#### 1) BEV ENCODER

The BEV encoder processes the transformed BEV features generated by the CBT block. As the feature map from the CBT block module has a resolution  $1/3\times$  of the final BEV resolution, a key role of the BEV encoder is to upscale the BEV feature maps for BEV representation. To integrate vehicle segmentation into our system, we simplified the segmentation process. We used a series of convolutions specific to the BEV. These convolutions consist of four  $3 \times 3$  filters, followed by a  $1 \times 1$  filter. This arrangement creates a BEV tensor with dimensions  $h \times w \times n$ , where  $n$  represents the number of categories. We set  $n$  to 1 because our focus is solely on vehicles and related elements, similar to the approach taken by [13]. To enhance the identification of roads and vehicles in our dataset, we employed an encoder-decoder transformer along with a specific equation

$$X = CBT(A1, A2) \quad (2)$$

In simpler terms,  $X$  represents the final segmentation map we aim to achieve.  $A1$  is the input image from one sensor modality, such as a camera, and  $A2$  is the input image from another sensor modality, such as map information. The function  $CBT(\cdot)$  is a transformer that combines information from both sensors to create a more precise segmentation map. The cross-attention mechanism, which helps in merging information from different modalities, can be applied using the following equation [71]:

$$M = \text{softmax} \left( \frac{Q \cdot (K^T)}{\sqrt{d_k}} \right) V \quad (3)$$

In this equation,  $Q$ ,  $K$ , and  $V$  represent the standard query, key, and value for attention mechanisms of transformer architectures [71] of each modality. The dot product between the query and the key denoted as  $Q \cdot (K^T)$ , is scaled by dividing it by the square root of the dimension of the key vector ( $d_k$ ) to prevent the dot product from becoming too large. The resulting attention weights are then used to weight the values associated with each modality, and these weighted values are combined to generate the output feature map  $M$ .

#### 2) SEGMENTATION HEADS

The semantic segmentation heads of the network are built on the BEV features map, focusing on tasks like vehicle and road layout. Each segmentation head shares a similar structure but varies in the number of output channels in the final layer. The task-specific head employs a ResNet-like block to extract features across different scales, combining these feature maps through upsampling. Subsequently, the BEV features map, with dimensions  $h_{bev} \times w_{bev} \times 256$ , is processed through several convolutional layers to generate the final output. The resulting segmentation map has dimensions  $h_{bev} \times w_{bev} \times 1$ , where each pixel value represents the probability that a vehicle or road surface occupies the corresponding BEV grid cell. As we demonstrate later in Table 5, we achieve state-of-the-art segmentation results by integrating the segmentation head with the trajectory prediction.

### D. GRAPH CONVOLUTIONAL NETWORK

Within the domain of ego vehicle trajectory prediction, the integration of GNNs represents a significant enhancement [14]. This approach builds upon prior research efforts that have explored similar methodologies [54], [72]. We integrate graph neural networks into the trajectory prediction problem as a novel step. In summary, we propose to build a graph using the motion agent and road positions (including the host vehicle) as the locations of the nodes of the graph. Adjacent nodes are connected, and the inverse of the distance between the nodes determines the weight of the connection. A GNN is used to process the graph representation. This lends spatial reasoning to the task of trajectory prediction. More formally, the primary aim behind incorporating GNNs in this context is to produce a holistic graph embedding that captures spatial information important for understanding and predicting future movement patterns. This goal is accomplished by leveraging object-feature interactions and their spatial proximity within a given scene. In this section, we provide some background on how graph neural networks operate in the context of our proposal.

#### 1) NOTATION

Consider a graph  $G = (V, E)$ , where  $V$  represents a set of nodes and  $E \subseteq V \times V$  represents a set of edges. Each node  $v \in V$  is associated with a feature vector  $x_v \in \mathbb{R}^{d_x}$ , and an edge between nodes  $u$  and  $v$  is indicated by  $(u, v) \in E$ . The neighborhood of a node  $v$  in the input graph, denoted by  $N(v)$ , consists of all nodes  $u$  connected to  $v$  by an edge. GNNs have become relatively popular for learning from data structured as graphs. They expand upon traditional neural network ideas to work with graphs, which helps in effectively handling relationships and structures [57] within the data. GNNs work by utilizing the connections and characteristics of nodes in a graph, which makes them well-suited for tasks like classifying nodes, predicting links, and classifying graphs. A key aspect of GNNs is how they gather information from neighboring nodes for each node. The core principle behind

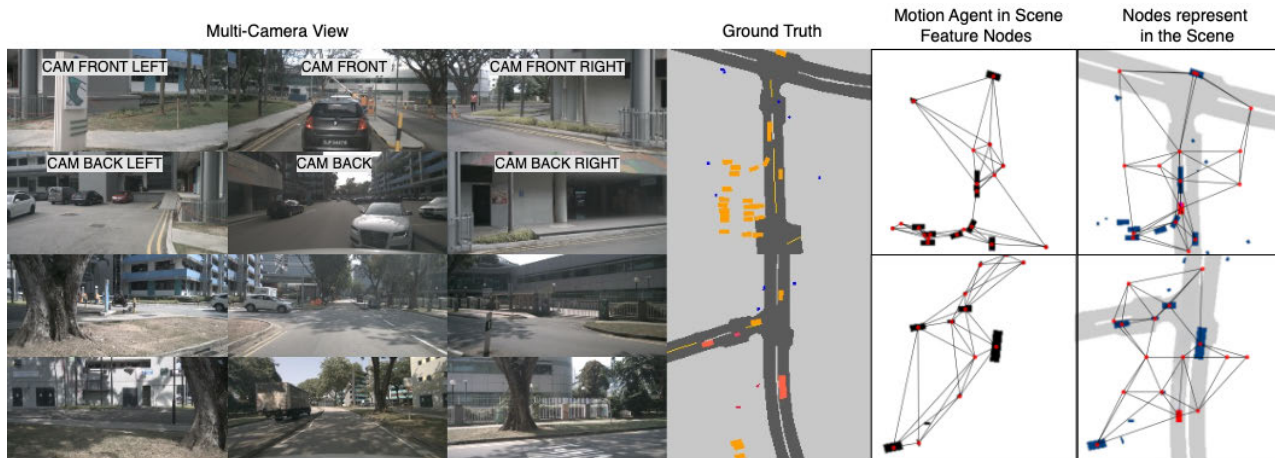


FIGURE 4. The figure on the left showcases multi-view camera images from the nuScene dataset alongside their corresponding ground truth annotations. On the right, motion agents are depicted as graph nodes, with the last image illustrating nodes representing roads and motion agents.

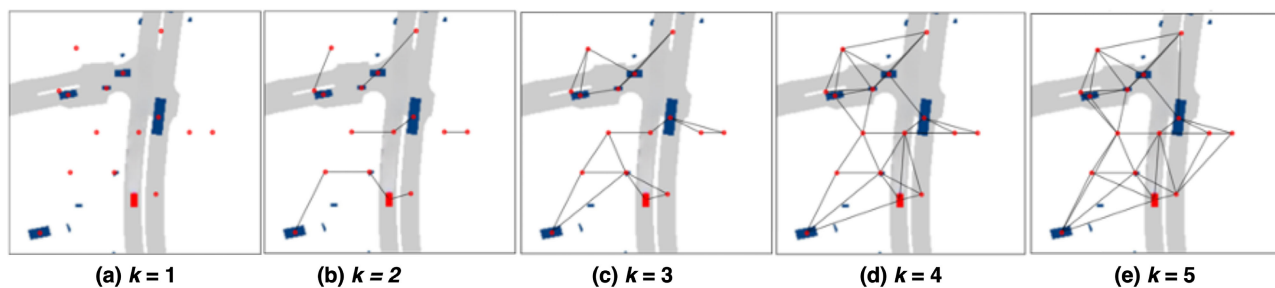


FIGURE 5. Exploring how changes in the parameter  $k$ , representing the number of nearest neighbors analyzed in the  $k$ NN algorithm, impact the connectivity and layout of nodes and edges in a graph. In the visualization, the red box indicates the ego vehicle, the blue boxes represent other agents in the scene, and the red dots denote the nodes.

this is that each node in a GNN updates its representation by considering information from its neighbors. The fundamental update rule for a node  $v$  in a GNN is represented as:

$$h_v^{(k+1)} = \text{UPDATE} \left( h_v^{(k)}, \text{AGGREGATE} \left( \{h_u^{(k)} : u \in \mathcal{N}(v)\} \right) \right) \tag{4}$$

$h_v^{(k)}$  signifies the feature vector for node  $v$  during the  $k$ -th iteration. The set of neighbors for  $v$ , denoted as  $\mathcal{N}(v)$ , is involved in the process. The functions AGGREGATE and UPDATE collect the features from the neighborhood and update the node’s features respectively [73]. In the context of our proposal, the AGGREGATE function gathers information from connected nodes (being motion agents in the neighborhood of the host vehicle) in the graph, such as position and distance, and is used to UPDATE the value of the target node. This iterative process enables each node to gather information from its immediate surroundings, progressively acquiring the ability to understand complex functions that reflect the graph’s structure.

In Figure 4, on the left side, we present multi-view camera images from the nuScenes dataset alongside their corresponding ground truth images. On the right side, we illustrate how the behavior of only the motion agents

in the scene changes when we consider feature nodes. This means that the agents in the scene are connected with  $(V, E)$ , where  $V$  represents vertices or nodes, and  $E$  represents edges. The BEV includes the road layout, the ego vehicle, and other objects in the scene. The nodes in the graph represent important objects such as road segments, drivable areas, pedestrians, and sidewalks. The edges show the relationships based on how close the objects are and how similar their features are.

TABLE 2. The table provides a detailed overview of the hyperparameters utilized in the GNN model, offering insight into the specific configurations and settings employed for the trajectory prediction task.

| S.No | Parameters Name      | Optimal Value |
|------|----------------------|---------------|
| 1    | GNN Input Dimension  | 1024          |
| 2    | GNN Hidden Dimension | 64            |
| 3    | Learning rate        | 0.001         |
| 4    | Weight Decay         | 0.001         |
| 5    | $k$ NN ( $k$ )       | 5             |
| 6    | Nodes                | 15            |

## 2) GRAPH CREATION

The main goal of the graph construction process is to enhance the spatial features within the scene, encode their relational attributes, and then learn them through training. To achieve

this, we leverage the characteristics of each object and its position coordinates within the image frame, information that is embedded in the graph structure. Our approach utilizes the  $k$ NN algorithm [17] to determine the neighborhood information of each object in the scene, facilitating the creation of connections between objects via edges. It is worth noting that these edge weights are calculated using the inverse of the Euclidean distance between objects [74].

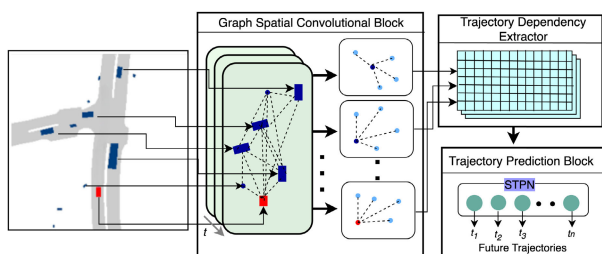
$$w_{ij} = \frac{1}{d_{ij}} \quad (5)$$

When the distance between two objects  $i$  and  $j$  is represented as  $d_{ij}$ , the weight of the edge connecting them (denoted as  $w_{ij}$ ) is calculated by (5).

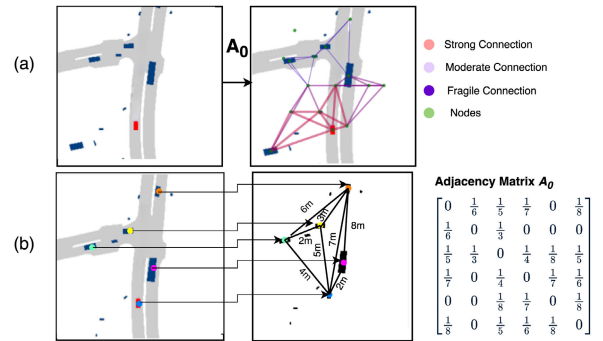
This method offers a significant advantage: by inversely correlating edge weights with distance, it ensures that closer objects have stronger connections in the graph, as illustrated in Figure 7(a). This resembles many real-world situations where proximity often suggests stronger interaction or similarity. Hence, this method is particularly effective for accurately representing spatial relationships and interactions within a scene. In Table 2, we present a detailed overview of the hyperparameters considered for GNN models.

In Figure 5, we examine how adjusting the parameter  $k$  affects the graph's connections between nodes and edges. This parameter  $k$  represents the number of nearest neighbors that the  $k$ NN algorithm considers when analyzing connectivity. For our  $k$ NN setup, we chose to set  $k = 5$ . This decision was made to ensure that sufficient connections were established between nodes and edges in the graph.

However, it is important to note that if you increase the value of  $k$  beyond 5, you might encounter a situation where edges start to overlap. This can pose challenges, especially when it comes to embedding the graph  $G$  into a lower-dimensional space for further analysis or visualization. Therefore, while a higher value of  $k$  may seem appealing for capturing more local structure in the data, it is crucial to strike a balance to prevent edge overlap and maintain the integrity of the graph representation.



**FIGURE 6.** The Graph-Based Spatial Convolutional Network demonstrates how it integrates vehicle information through a Graph Spatial Convolutional Block. This block captures spatial relationships between nearby objects, enhancing the model's understanding of the environment. Simultaneously, the network extracts temporal features to analyze how trajectories evolve over time. Finally, the spatial features obtained are fed into a trajectory prediction model to accurately predict the ego vehicle's trajectory. (Note: in this case,  $t_i$  refers to trajectory, rather than camera calibration parameters.)



**FIGURE 7.** (a) This demonstrates a quantitative representation of the spatial relationships among all motion agents in the graph, categorized into strong, moderate, and fragile connections. (b) An example illustrating graph generation at a specific timestamp, where each vehicle's color corresponds to the color of the respective vertex in the graph. Notably, the weights in the adjacency matrix  $A_0$  are annotated as reciprocals of distances.

### 3) GRAPH-BASED SPATIAL REPRESENTATION AND ADJACENCY MATRIX

In Figure 6, we draw inspiration from the graph's topological structure and how it, along with our model, aids in understanding the spatial relationships among different vehicles. The spatial graph represents the logical relationships between the positions of objects. For example, it helps understand which objects are neighbors, the distance between them, and how they are connected.

The Graph-based spatial representation is defined as  $G_t = (V_t, E_t)$ , where  $V_t$  is the set of all vertices, denoted by  $v_t^n$  for each vehicle, with  $n$  ranging from 1 to  $N$ . Essentially, each  $v_t^n$  represents an individual vehicle in the scene. On the other hand,  $E_t$  represents the set of all edges in the graph. These edges symbolize the interactions or mutual effects between vehicles, as depicted in Figure 6. Our study explored  $A_t \in \mathbb{R}^{N \times N}$  as a weighted adjacency matrix. The entries in this matrix indicate the strength of connections between vehicles, ranging from strong to moderate and fragile. We observed that objects close tend to have stronger bonds, as depicted in Figure 7 (a). To quantify this relationship, we used the reciprocal of the distance as a measure of weight between motion agents in the scene. This ensures that closer objects are assigned higher weights. An example of how a spatial graph is generated is illustrated in Figure 7 (b), where each vertex represents a vehicle, and the weight in the weighted adjacency matrix  $A_0$  is determined by the reciprocal of the distance.

### 4) POSITIONAL ENCODING OF NODES IN GRAPH

In our graph, we incorporate the attributes of objects and nodes and their distances. To assist the model in understanding the spatial relationships among objects within the scene, we incorporate positional encoding into the nodes [75]. This approach ensures that features with comparable characteristics are grouped closely together.

Given a dimension  $\text{dim}$  and a maximum length  $\text{max\_len}$ , the positional encoding matrix PE is computed as follows:

$$\begin{aligned} \text{PE}_{(i,2j)} &= \sin\left(i \cdot 10^{-\frac{2j}{\text{dim} \cdot \log(10000)}}\right) \\ \text{PE}_{(i,2j+1)} &= \cos\left(i \cdot 10^{-\frac{2j}{\text{dim} \cdot \log(10000)}}\right) \end{aligned} \quad (6)$$

The vector  $i$  consists of values ranging from 0 to  $\text{max\_len} - 1$ , indicating the position within the sequence, while  $j$  ranges from 0 to  $\frac{\text{dim}}{2} - 1$ . PE is the positional encoding matrix where each row corresponds to a position, and each column corresponds to a dimension of the encoding. This enhanced node feature is then further used for downstream tasks.

### E. SPATIO-TEMPORAL PROBABILISTIC NETWORK

This section describes a novel step in integrating STPNs in the BEV framework for trajectory prediction. We use graph embeddings as input to an STPN to help our model better predict the likely trajectory for the ego vehicle. The objective of this network is to predict the future states of the ego vehicle and a detailed map, assuming we have high-quality state outputs from an object detection and tracking system suitable for autonomous vehicles, as per [76], [77].

The set  $\mathcal{N}_t$  represents the agents the ego vehicle interacts with at time  $t$ , and  $\mathbf{x}_t^i$  is the state of agent  $i \in \mathcal{N}_t$  at time  $t$ . The trajectory of agent  $i$  from times  $t_1 = (t_1, \dots, t_n)$  is denoted by  $\mathcal{Z}_{t_1:t_n}^i = [\mathbf{x}_{t_1}^i, \dots, \mathbf{x}_{t_n}^i]$ , where  $t_1 < t_n$  and  $i \in \mathcal{N}_t$ .

Furthermore, we assume that the detailed map will be available as outlined in our method. This includes lane layouts, crosswalks, drivable areas, and other relevant data. Map-level information is provided in the nuScenes dataset and can be considered a specific case of High-Definition (HD) Maps. HD Maps are considered to be an important part of progress towards vehicle automation [78], [79], with many localization methods utilizing map information [80]. For trajectory estimation, it is common in research to use map information, e.g., CoverNet [63], Trajectron++ [81], and MultiPath [15] (later, in Section IV-C, we will use these as baselines to compare against our proposed method). The context of the scene over the past  $T$  steps, encompassing the map and the partial history of the ego vehicle, is represented by  $C = \{\bigcup_i \mathcal{Z}_{t-t_1:t}^i; \text{Map Data}\}$ .

Our approach integrates a trajectory prediction layer, following the method introduced in [15]. To enhance performance in this area, we utilize ResNet-50 [16], as suggested by prior research [15], [76]. While our network currently makes predictions for one agent at a time, it has the potential to predict multiple agents simultaneously, similar to [76]. However, we focus on single-agent predictions (as in [15]) to simplify the paper and highlight our main contributions. To depict probabilistic trajectory predictions in various scenarios, we apply a classification technique that selects the appropriate trajectory set based on the agent of interest and the scene context  $C$ . The softmax distribution is used and is consistent with classification practices. Specifically, the

likelihood of the  $k$ -th trajectory is calculated as follows

$$\mathbf{P}(\mathcal{Z}_{t:t+N}^k | x) = \frac{\exp(f_k(x))}{\sum_i \exp(f_i(x))} \quad (7)$$

We calculate the likelihood of a sequence  $\mathcal{Z}_{t:t+N}$  given the input  $x$ . Here,  $\mathbf{P}$  stands for the probability function,  $\mathcal{Z}_{t:t+N}$  is the sequence we are analyzing, and  $x$  is the input data. This calculation employs the softmax function on the output of a neural network. Specifically,  $f_k(x)$  is the output for a sequence  $k$ , and  $\sum_i \exp(f_i(x))$  is the sum of the exponentiated outputs for all sequences. We have adapted the Multi-Trajectory Prediction (MTP) method [15] for our datasets. This model predicts a fixed number of trajectories (modes) and their corresponding probabilities. However, our current emphasis is on single trajectory prediction (STP) [82].

### F. ARCHITECTURE OVERVIEW

As the overall architecture is perhaps a little bit complex, it is worth taking a moment to summarise briefly. In the inference phase, our proposed method (BEVseg2GTA) is executed end-to-end for jointly segmenting vehicles, and predicting the ego vehicle's trajectory involves a series of steps. First, features at various scales are extracted by a backbone (in this case, EfficientNet-B4, with the scaling being one of the key innovations of EfficientNet [83]) from images at various scales. Next, map-view positional embedding and cross-attention layers are used to collect the features from different angles. Then the feature space is converted to a top-view by projecting features from the image space to BEV space, where semantic segmentation of the related agents is performed (which, as we shall see in Table 6 (right-most grey column), provides state-of-the-art segmentation results).

We then propose two novel steps. The first is that the segmentation results are processed through a  $k$ NN algorithm to represent each object as a node in a graph, followed by a GNN to create an embedding that represents the environment around the ego vehicle. Finally, these embeddings are used as input to a spatio-temporal probabilistic layer for predicting the future trajectory of the ego vehicle, taking into account the context provided by the surrounding scene. This inference framework is depicted in Figure 3.

## IV. EXPERIMENTAL SETUP AND EVALUATIONS

### A. DATASET

To evaluate our method, we considered the nuScenes dataset [18], which comprises 1000 video sequences captured in Boston and Singapore. Each video is 20 seconds long and contains 40 frames, resulting in a total of 40,000 samples. The dataset is divided into training, validation, and testing sets, each containing 700, 150, and 150 scenes, respectively. The collected data offers a 360° view of the autonomous vehicle's surroundings, captured from six different camera angles. For consistency, we adopted the same train-validation-test split as in previous studies [13], [37].

## B. EVALUATION METRICS

We calculate the evaluation metrics for our model by comparing it with state-of-the-art models, employing a comprehensive set of performance measures to assess its effectiveness and robustness, as done in previous studies [15], [63], [76], [77], [81].

*IoU*: The well-known Intersection over Union (IoU) metric is used in this work to estimate the performance of semantic segmentation tasks for vehicles and pedestrians.

*MinADE<sub>h</sub>*: We compute the minimum average and final displacement errors across  $h$  predicted trajectories for ego vehicle trajectory prediction, where the minimum over  $h$  avoids penalizing the model for generating plausible future trajectories that do not correspond to the ground truth.

$$\text{MinADE}_h = \min_{i \in \{1 \dots h\}} \frac{1}{T_f} \sum_{t=1}^{T_f} \|y_t - \hat{y}_t^{(i)}\|_2 \quad (8)$$

where  $y_t$  denotes the true position at time  $t$ , and  $\hat{y}_t^{(i)}$  represents the estimated position at time  $t$  for the  $i^{\text{th}}$  predicted trajectory (out of a total of  $h$ ).  $h$  is a parameter represents the number of predictions considered, with  $h$  set to 5, 10, or 15 in our case. The variable  $T_f$  refers to the total number of time steps over which the average error is computed.

*MinFDE<sub>h</sub>*: We evaluate the minimum final displacement error across  $h$  predicted trajectories for ego vehicle trajectory prediction. *MinFDE<sub>h</sub>* is similar to *MinADE<sub>h</sub>*, but it only evaluates the prediction error at the final predicted location.

$$\text{MinFDE}_h = \min_{i \in \{1 \dots h\}} \|y_{T_f} - \hat{y}_{T_f}^{(i)}\|_2 \quad (9)$$

*MissRate<sub>h,d</sub>*: In [77], the authors also introduced the *MissRate<sub>h,d</sub>* (considering a given distance threshold  $d$  and the  $h$  most likely predictions generated by the STPN model), which computes the fraction of missed predictions.

## C. BASELINES

We compare our model to seven baseline methods, two physics-based approaches, and five recently proposed models that represent the state of the art for predicting the trajectories of ego vehicles.

- **Constant Velocity and Yaw** is the most basic baseline we use. It is a physics-based model that calculates the future trajectory while assuming the vehicle's velocity and yaw rate are constant and based on its current state.
- **Physics oracle** [63] is an extension of classic physics-based models. It uses the current vehicle state (velocity, acceleration, and yaw) to determine the minimum average point-wise Euclidean distance between predictions.
- **CoverNet** [63] approaches multimodal trajectory prediction only as a classification problem. They predict the probability of a set of fixed trajectories based on the state of the target vehicle.
- **Trajectron++** [81] is an LSTM with a graph-based architecture that predicts the trajectories of agents while considering agent motion and diverse scene data.

- **MTP** [15] utilizes a CNN-based approach to encode the surrounding context of each agent into a raster image and predicts a set of probable trajectories.
- **MultiPath** [76] employs a set of anchors to estimate the distribution of predicted trajectories, which generates a Gaussian mixture at each time step  $t$ .
- **MHA-JAM** [77] utilizes multi-head attention with a joint representation of the static scene and agents. It calculates probable trajectories along with their corresponding attention maps.

## D. IMPLEMENTATION DETAILS

To build our architecture, we use a fine-tuned EfficientNet-B4 model. This model resizes images to two dimensions: (28, 60) and (14, 30), corresponding to  $8 \times$  and  $16 \times$  downscaling, respectively. The initial map view utilizes a parameter tensor of size  $w \times h \times D$ , with  $D$  set to 128. To manage computation efficiently, we set  $w = h = 25$  as the complexity of the cross-attention function increases with grid size.

The encoder features two cross-attention blocks, each handling a scale of features. These blocks use multi-head attention with four heads and an embedding size of  $d_{head} = 64$ . The map-view representation obtained from the cross-attention transformer then undergoes joint vehicle segmentation to accurately identify vehicle segments as BEV features.

Following this, the decoder consists of three layers, each combining bilinear upsampling with convolution. These layers progressively increase the resolution of the latent representation, doubling it each time until reaching a final output size of  $200 \times 200$ . This resolution corresponds to a  $100 \times 100$  meter area centered around the ego vehicle.

Then, the BEV representation is input into the  $k$ NN algorithm to determine each object's neighborhood and establish spatial relationships. We set  $k$  to 5 to capture sufficient spatial relationships between objects in the scene (see Figure 5). The  $k$ NN algorithm generates node features, which are used by a neural network to create embeddings of the environment. These embeddings are then utilized as input to the Spatial-Temporal Probabilistic Network (STPN), which offers probabilistic predictions. Instead of providing a single deterministic trajectory, the network offers a probability distribution over possible future trajectories (see Figure 3). This information aids in identifying the motion planning of the ego vehicle. Specifically, while the graph embeddings include both image and node features, we only use the node features related to the ego vehicle. This approach allows the system to more accurately estimate the ego vehicle's position, speed, and orientation relative to other objects in the environment.

*Training Stages*: Firstly, for the BEV segmentation, we utilized both focal loss [84] and binary cross-entropy [85] as separate loss functions, as detailed in Table 3. Training was conducted over 30 epochs with a batch size of 4. Optimization was achieved using the AdamW optimizer [86], with a learning rate set to  $1e-2$  and a weight decay of  $1e-7$ .

Secondly, the GNN model was trained using the mean square error (MSE) loss function. This phase involved a batch size of 16 over 50 epochs, with optimization performed using the Adam optimizer [87], a learning rate of 0.001, and a weight decay of 0.001. Finally, the STPN module was trained with a batch size of 32, utilizing the Adam optimizer [87]. Note that all implementations were conducted using PyTorch [88].

### E. ABLATION STUDY

We conduct a detailed ablation study to gain insights into the performance and functionality of our model.

#### 1) SEGMENTATION LOSS ANALYSIS

In our ablation study for the segmentation task using the nuScenes dataset, we evaluated the performance of two loss functions: Binary Focal Loss and binary cross Entropy. Both loss functions were tested with a binary classification problem, having two classes. The results, as presented in Table 3, indicate that Binary Cross Entropy achieved a lower loss value of 0.1848 compared to Binary Focal Loss, which had a value of 0.2758. This demonstrates that Binary Cross Entropy performs better for our specific segmentation task, making it the preferred choice over binary Focal Loss.

**TABLE 3.** Different loss functions were employed in this ablation study for the segmentation task using the nuScenes dataset [18].

| Loss Function             | No. of Class | Loss ↓ |
|---------------------------|--------------|--------|
| Binary Focal Loss [84]    | 2            | 0.2758 |
| Binary Cross Entropy [85] | 2            | 0.1848 |

#### 2) GRAPH NEURAL NETWORK WITH HIDDEN UNITS

In this task, we chose appropriate hyperparameters to improve the performance of our network. In Table 4, we compare the Mean Squared Error (MSE) of our model with varying numbers of hidden units in the GNN, ranging from {16, 32, 64, 100, 128}. Note that for this task, we only used Vision Transformer (ViT) [89] as the backbone with a feature size of 1024.

**TABLE 4.** Comparison of Mean Squared Error (MSE) for GNN models with varying numbers of hidden units, utilizing ViT [89] as the backbone and featuring a size of 1024.

| Prediction Horizon (Sec) | 16   | 32   | 64          | 100  | 128         |
|--------------------------|------|------|-------------|------|-------------|
| 1                        | 0.62 | 0.60 | <b>0.48</b> | 0.48 | 0.69        |
| 2                        | 0.87 | 0.76 | 0.63        | 0.76 | <b>0.51</b> |
| 3                        | 0.98 | 0.54 | <b>0.35</b> | 0.37 | 0.39        |
| 4                        | 1.77 | 0.98 | 0.95        | 0.36 | <b>0.29</b> |
| 5                        | 2.72 | 1.97 | <b>1.19</b> | 1.93 | 1.45        |
| 6                        | 3.13 | 2.09 | <b>2.03</b> | 2.43 | 2.93        |
| Average ( $\bar{x}$ )    | 1.68 | 1.15 | <b>0.93</b> | 1.06 | 1.04        |

#### 3) ABLATION STUDY ON DIFFERENT SETTINGS IN BEV SPACE

We evaluate the performance of our model in generating BEV map representations and planning trajectories using the publicly available nuScenes dataset. This evaluation

encompasses two distinct settings - ‘Setting 1’, characterized by a grid size of  $100m \times 50m$  with a resolution of  $25cm$ , and ‘Setting 2’, which employs a grid size of  $100m \times 100m$  with a resolution of  $50cm$ . Table 5 provides a comparison of our proposed approach with other existing methods, including LSS [37] and CVT [13], where R is the resolution of the BEV representation.

**TABLE 5.** We compare the Intersection-over-Union (IoU) metrics for segmenting vehicles in the nuScenes dataset [18]. ‘Setting 1’ uses a grid measuring  $100m \times 50m$  with a resolution of  $25cm$ , while ‘Setting 2’ uses a grid of  $100m \times 100m$  with a resolution of  $50cm$ . For further analysis, please refer to the references for LSS [37] and CVT [13].

| Method      | Setting 1   | Setting 2   | #Params (M) | FPS       | R        |
|-------------|-------------|-------------|-------------|-----------|----------|
| LSS [37]    | -           | 32.1        | 14          | 25        | -        |
| CVT [13]    | 37.5        | 36.0        | 5           | 35        | 2        |
| <b>Ours</b> | <b>37.8</b> | <b>37.9</b> | <b>5</b>    | <b>35</b> | <b>1</b> |

## V. RESULTS

In this section, we present our experimental results. For clarity, we have divided the results into quantitative and qualitative categories. Finally, we provide an analysis of specific failure cases of our method.

### A. QUANTITATIVE ANALYSIS

Firstly, we evaluate the BEV segmentation performance using different methods, including LSS [37] and CVT [13], and compare them with our model. The grey column ‘‘Vehicle’’ represents the vehicle segmentation performance of each method. The values in this column indicate how well each method performs at segmenting vehicles, with higher values indicating better performance. Our experiments demonstrate promising results when employing a resolution of  $R = 1$ . However, increasing  $R$  to 2, as suggested by CVT (see Table 5), may result in the loss of information in the camera-view representation of each input image, particularly BEV features. Further evaluations are conducted using various methods, as outlined in Table 6

Secondly, we perform a comprehensive experimental ablation study, evaluating it against four baseline models [15], [63], [76], [81]. These baselines are a recently proposed model considered the current state-of-the-art model for trajectory prediction. We conducted this comparison to assess how effectively and accurately our model predicts trajectories compared to existing models. We aim to ascertain whether our model outperforms or matches the performance of current baseline models. This assessment helps us understand the strengths and weaknesses of our model and identifies areas for potential improvement. To evaluate our model’s performance on the nuScenes dataset, we obtained the output trajectory  $[y_1, y_2, y_3, \dots, y_n]$ .

We evaluated the model’s performance on the dataset for different values of  $h$ , where  $h$  is set to 5, 10, and 15. To compute  $\text{MinADE}_h$  and  $\text{MinFDE}_h$ , we use equations (8) and (9). To train the model, we minimized the minimum average displacement error over  $h$  ( $\text{MinADE}_h$ ) on the training set, as shown in Table 7. In other words, we aimed to reduce

**TABLE 6.** We compare the performance of different methods for vehicle segmentation on the nuScenes dataset [18], including our proposed approach. This comparison utilizes Intersection over Union (IoU) scoring, focusing on the BEV segmentation task. The right-most column highlights instances where vehicle segmentation has improved in the presented results.

| Method      | Surround-View Camera | Input Image Size (px) | Feature Extractor | Grid Scale/ Unit Size | FPS | Vehicle IoU (%) ↑ |
|-------------|----------------------|-----------------------|-------------------|-----------------------|-----|-------------------|
| PanSeg [90] | ×                    | 448 × 768             | EfficientDet      | -                     | -   | 35.06             |
| GitNet [91] | ×                    | -                     | ResNet50          | 200 × 200 / 0.25m     | -   | 35.90             |
| M2BEV [44]  | ✓                    | 900 × 1600            | ResNeXt-101       | 200 × 200 / 0.5m      | -   | -                 |
| LSS [37]    | ✓                    | 128 × 352             | EfficientNet-B0   | 200 × 200 / 0.5m      | 25  | 32.1              |
| CVT [13]    | ✓                    | 200 × 200             | EfficientNet-B4   | 200 × 200 / 0.5m      | 35  | 36.0              |
| CoBEVT [43] | ✓                    | 200 × 200             | EfficientNet-B4   | 200 × 200 / 0.5m      | 35  | 37.1              |
| <b>Ours</b> | ✓                    | 200 × 200             | EfficientNet-B4   | 200 × 200 / 0.5m      | 35  | <b>37.9</b>       |

**TABLE 7.** Evaluation of competing methods on the nuScenes dataset, analyzing the Minimum Average Prediction Error (MinADE) and Final Displacement Error (MinFDE) over a 6-second prediction horizon.

| Method             | MinADE <sub>5</sub> ↓ | MinADE <sub>10</sub> ↓ | MinADE <sub>15</sub> ↓ | MinFDE <sub>5</sub> ↓ | MinFDE <sub>10</sub> ↓ | MinFDE <sub>15</sub> ↓ | MissRate <sub>5,2</sub> ↓ | MissRate <sub>10,2</sub> ↓ |
|--------------------|-----------------------|------------------------|------------------------|-----------------------|------------------------|------------------------|---------------------------|----------------------------|
| Const. Vel and Yaw | 4.61                  | 4.61                   | 4.61                   | 11.21                 | 11.21                  | 11.21                  | 0.91                      | 0.91                       |
| Physics oracle     | 3.69                  | 3.69                   | 3.69                   | 9.06                  | 9.06                   | 9.06                   | 0.88                      | 0.88                       |
| CoverNet [63]      | 2.62                  | 1.92                   | 1.63                   | 11.36                 | -                      | -                      | 0.76                      | 0.64                       |
| Trajectron++ [81]  | 1.88                  | 1.51                   | -                      | -                     | -                      | -                      | 0.70                      | 0.64                       |
| MTP [15]           | 2.22                  | 1.74                   | 1.55                   | 4.83                  | 3.54                   | 3.05                   | 0.74                      | 0.67                       |
| MultiPath [76]     | 1.78                  | 1.55                   | 1.52                   | <b>3.62</b>           | 2.93                   | 2.89                   | 0.78                      | 0.76                       |
| MHA-JAM [77]       | 1.85                  | <i>1.24</i>            | <b>1.03</b>            | 3.72                  | 2.23                   | <i>1.67</i>            | <i>0.60</i>               | <b>0.46</b>                |
| <b>Ours</b>        | <b>1.63</b>           | <b>1.19</b>            | <i>1.06</i>            | <i>3.63</i>           | <b>2.13</b>            | <b>1.65</b>            | <b>0.56</b>               | <i>0.51</i>                |

Best is marked in **bold**, second best is marked in *italics*

the error between the predicted and actual trajectories by minimizing the minimum distance between the two for each of the  $h$  time steps. This method allowed us to improve the accuracy of our model's predictions and ensure that it performs well on the nuScenes dataset.

## B. QUALITATIVE ANALYSIS

In Figure 8, we compare two current models for vehicle segmentation: CVT [13] and LSS [37]. These models have limitations as they do not capture enough information for accurate vehicle segmentation, indicated by black and pink colors, respectively. Our proposed method addresses these issues by providing a comprehensive view and sufficient information for accurate vehicle segmentation. Additionally, our model's improved segmentation capability enhances vehicles' visual representation and leads to more accurate ego vehicle trajectory predictions. A better understanding of the surrounding vehicles allows for more precise path planning.

As shown in Figure 8, the image's left side features six camera angles around a vehicle, with the top three looking forward and the bottom three looking backward. The right side of the image provides the ground truth segmentation for comparison. Moving from right to left, the second image presents our trajectory prediction and enhanced map-view segmentation for vehicles and drivable areas. The third image introduces the LSS method [37], and the fourth image showcases the CVT method [13], facilitating a fair comparison and result presentation.

This figure contrasts the segmentation outcomes from three models for vehicles and roads. The visualization uses **black** for CVT results, pink for LSS results, red for our model's results, and blue for the nuScenes ground truth, which is the actual image segmentation. This helps to give an intuitive understanding of the CVT and LSS models' performance against our proposed model.

## C. FAILURE CASE ANALYSIS IN PREDICTING EGO VEHICLE TRAJECTORY

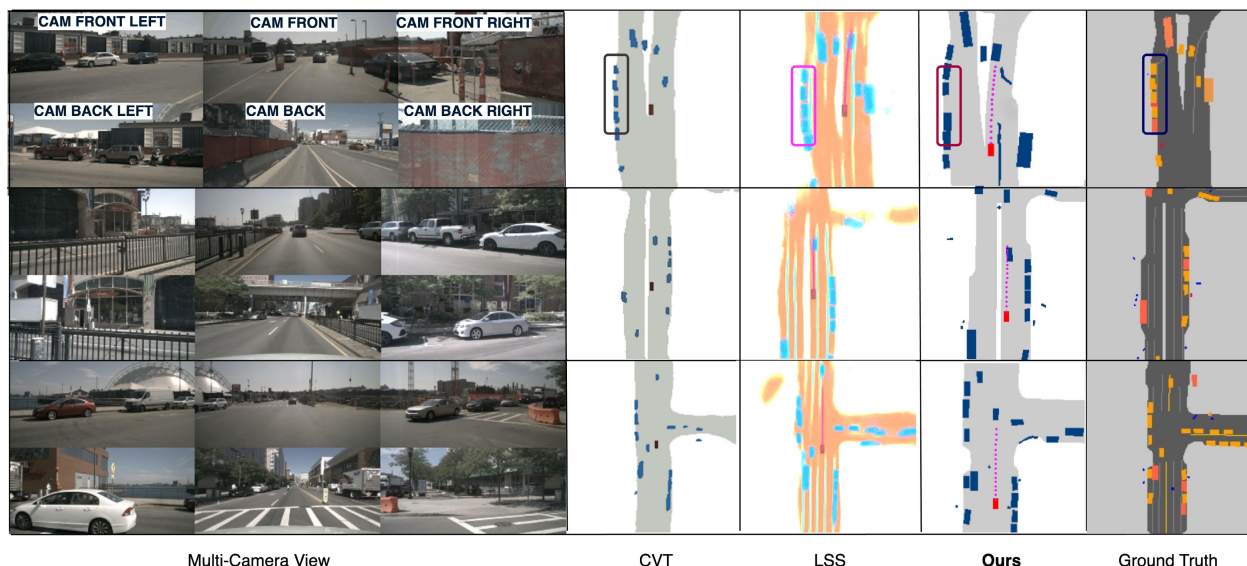
This section discusses some select failure cases encountered in our model. Figure 9 illustrates a failed ego vehicle trajectory prediction scenario from the nuScenes dataset. On the left side, multi-view camera images from the nuScenes dataset are presented alongside their corresponding ground truth annotations.

On the right side, two failure scenarios are highlighted as follows:

- 1) The first upper scenario depicts a lack of a predicted trajectory. This failure occurs when a large bus blocks the ego vehicle, preventing our model from calculating the trajectory. However, once the ego vehicle gains enough space from the front vehicle, our model can successfully predict the ego vehicle's trajectory. We highlight the first failure case in **violet** color for better visibility.
- 2) The second bottom scenario illustrates a trajectory intersecting with another agent in the scene. This failure arises from the model's inability to predict behavioral unpredictability, meaning it struggles to anticipate the unpredictable behavior of other agents. For better visibility, we highlight the second failure case in **cyan** color.

## D. DISCUSSION AND LIMITATIONS

The qualitative and quantitative results presented demonstrate that our proposed model appears to improve upon both CVT and LSS (and other segmentation models) in vehicle and road segmentation tasks. Our model more accurately locates vehicles in vehicle segmentation, while CVT and LSS appear less precise. This is visible in Figure 8, where red markings (our model results) are closer to green markings (ground truth) than pink markings (LSS model results) and black



**FIGURE 8.** Qualitative outcomes of our model (BEVSeg2GTA): The six camera perspectives of nuScenes surrounding the vehicle are shown, with the top three facing forward and the bottom three facing backward. Ground truth segmentation is displayed on the right. Our trajectory prediction approach integrates improved map-view segmentation with ego vehicle trajectory (second from the right), and it is compared to the LSS method [37] and the CVT method [13] (third and fourth from the right).



**FIGURE 9.** An illustration showcases a failed prediction scenario in ego vehicle trajectory forecasting using data from the NuScenes dataset. On the left side, multi-view camera images from the NuScenes dataset are presented alongside their corresponding ground truth annotations. Two failure scenarios are highlighted on the right side: the first upper scenario depicts a lack of a predicted trajectory. In contrast, the second bottom scenario illustrates a trajectory intersecting with another agent in the scene.

markings (CVT model results). Similarly, our model captures more detailed road segmentation information than the CVT and LSS models. In parallel, our proposed method provides state-of-the-art results for trajectory prediction across most metrics (refer to Table 7). Our method achieves the lowest or second lowest errors across all categories.

Our experiments show promising results with a resolution of  $R = 1$  (per Table 5). When we go to a higher resolution by setting  $R$  to 2, we observe a degradation in segmentation performance. We can hypothesize that this decline can be attributed to the elevated placement of cameras, which causes objects within the scene to appear significantly smaller. As a result, crucial features are lost in the BEV representation.

We acknowledge that our model has limitations, as demonstrated in Figure 9. While it performs well when using a raster map image, it struggles to capture sufficient information between different types of agents or the complexities of urban environments when generating a BEV. This limitation leads to inaccuracies in ego vehicle trajectory prediction. Despite these limitations, they do not significantly impact our model’s overall performance.

## VI. CONCLUSION AND FUTURE WORK

This paper proposes BEVSeg2GTA, which utilizes a BEV generated from a multi-view camera, alongside joint vehicle segmentation and neural network graphs, to predict ego

vehicle trajectories. Our approach demonstrates promising results for reliable and efficient trajectory prediction, outperforming existing state-of-the-art methods. We employ an encoder-decoder transformer to process images captured by the multi-view cameras and convert them into BEV representations. We then use the BEV features in a GNN to establish spatial relationships between agents. With the obtained spatial information, we apply the  $k$ NN algorithm to generate a graph embedding, providing insights into the spatial structure of the scene. Furthermore, we integrate the graph embedding with an STPN to predict ego vehicle trajectories accurately. Finally, the predicted trajectories are visualized from a bird's-eye view of the ego vehicle, offering a comprehensive understanding of the surrounding traffic dynamics. Our study highlights the potential benefits of incorporating transformer-based models and GNNs into spatiotemporal networks, emphasizing their ability to enhance trajectory prediction accuracy significantly.

Future research aims to incorporate multi-modal sensors, such as LiDAR and radar, to enhance the BEV and trajectory prediction capabilities. Additionally, we plan to explore the multi-trajectory prediction of other vehicles. We also intend to examine the impact of camera calibration accuracy on the performance of BEV algorithms.

## REFERENCES

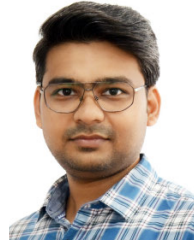
- [1] L. Claussmann, M. Revilloud, D. Gruyer, and S. Glaser, "A review of motion planning for highway autonomous driving," *IEEE Trans. Intell. Transp. Syst.*, vol. 21, no. 5, pp. 1826–1848, May 2020.
- [2] A. Chougule, V. Chamola, A. Sam, F. R. Yu, and B. Sikdar, "A comprehensive review on limitations of autonomous driving and its impact on accidents and collisions," *IEEE Open J. Veh. Technol.*, vol. 5, pp. 142–161, 2024.
- [3] S. Mozaffari, O. Y. Al-Jarrah, M. Dianati, P. Jennings, and A. Mouzakitis, "Deep learning-based vehicle behavior prediction for autonomous driving applications: A review," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 1, pp. 33–47, Jan. 2022.
- [4] J. Zhao, W. Zhao, B. Deng, Z. Wang, F. Zhang, W. Zheng, W. Cao, J. Nan, Y. Lian, and A. F. Burke, "Autonomous driving system: A comprehensive survey," *Expert Syst. Appl.*, vol. 242, May 2024, Art. no. 122836.
- [5] H. Liao, Z. Li, H. Shen, W. Zeng, D. Liao, G. Li, and C. Xu, "BAT: Behavior-aware human-like trajectory prediction for autonomous driving," in *Proc. AAAI Conf. Artif. Intell.*, vol. 38, 2024, pp. 10332–10340.
- [6] Y. Song, Q. Jiang, W. Chen, X. Zhuang, and G. Ma, "Pedestrians' road-crossing behavior towards eHMI-equipped autonomous vehicles driving in segregated and mixed traffic conditions," *Accident Anal. Prevention*, vol. 188, Aug. 2023, Art. no. 107115.
- [7] H. Shao, L. Wang, R. Chen, H. Li, and Y. Liu, "Safety-enhanced autonomous driving using interpretable sensor fusion transformer," in *Proc. Conf. Robot Learn.*, 2023, pp. 726–737.
- [8] X. Chen, H. Zhang, F. Zhao, Y. Hu, C. Tan, and J. Yang, "Intention-aware vehicle trajectory prediction based on spatial-temporal dynamic attention network for Internet of Vehicles," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 10, pp. 19471–19483, Oct. 2022.
- [9] C. Yang and Z. Pei, "Long-short term spatio-temporal aggregation for trajectory prediction," *IEEE Trans. Intell. Transp. Syst.*, vol. 24, no. 4, pp. 4114–4126, Apr. 2023.
- [10] M. Fu, T. Zhang, W. Song, Y. Yang, and M. Wang, "Trajectory prediction-based local spatio-temporal navigation map for autonomous driving in dynamic highway environments," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 7, pp. 6418–6429, Jul. 2022.
- [11] R. Quan, L. Zhu, Y. Wu, and Y. Yang, "Holistic LSTM for pedestrian trajectory prediction," *IEEE Trans. Image Process.*, vol. 30, pp. 3229–3239, 2021.
- [12] N. Dong, S. Chen, Y. Wu, Y. Feng, and X. Liu, "An enhanced motion planning approach by integrating driving heterogeneity and long-term trajectory prediction for automated driving systems: A highway merging case study," *Transp. Res. C, Emerg. Technol.*, vol. 161, Apr. 2024, Art. no. 104554.
- [13] B. Zhou and P. Krähenbühl, "Cross-view transformers for real-time map-view semantic segmentation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2022, pp. 13750–13759.
- [14] Y. Xu, L. Wang, Y. Wang, and Y. Fu, "Adaptive trajectory prediction via transferable GNN," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2022, pp. 6510–6521.
- [15] H. Cui, V. Radosavljevic, F.-C. Chou, T.-H. Lin, T. Nguyen, T.-K. Huang, J. Schneider, and N. Djuric, "Multimodal trajectory predictions for autonomous driving using deep convolutional networks," in *Proc. Int. Conf. Robot. Autom. (ICRA)*, May 2019, pp. 2090–2096.
- [16] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778.
- [17] P. Cunningham and S. J. Delany, "K-nearest neighbour classifiers—A tutorial," *ACM Comput. Surv.*, vol. 54, no. 6, pp. 1–25, Jul. 2022.
- [18] H. Caesar, V. Bankiti, A. H. Lang, S. Vora, V. E. Liong, Q. Xu, A. Krishnan, Y. Pan, G. Baldan, and O. Beijbom, "NuScenes: A multimodal dataset for autonomous driving," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 11618–11628.
- [19] S. Sharma, A. Das, G. Sistu, M. Halton, and C. Eising, "BEVSeg2TTP: Surround view camera bird's-eye-view based joint vehicle segmentation and ego vehicle trajectory prediction," in *Proc. 19th Int. Joint Conf. Comput. Vis., Imag. Comput. Graph. Theory Appl.*, 2024, pp. 25–34.
- [20] Z. Zhu, Y. Zhang, H. Chen, Y. Dong, S. Zhao, W. Ding, J. Zhong, and S. Zheng, "Understanding the robustness of 3D object detection with bird's-eye view representations in autonomous driving," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2023, pp. 21600–21610.
- [21] S. Wang, X. Zhao, H. Xu, Z. Chen, D. Yu, J. Chang, Z. Yang, and F. Zhao, "Towards domain generalization for multi-view 3D object detection in bird-eye-view," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2023, pp. 13333–13342.
- [22] L. Zheng, S. Li, B. Tan, L. Yang, S. Chen, L. Huang, J. Bai, X. Zhu, and Z. Ma, "RCFusion: Fusing 4-D radar and camera with bird's-eye view features for 3-D object detection," *IEEE Trans. Instrum. Meas.*, vol. 72, pp. 1–14, 2023.
- [23] M. Zhu, S. Zhang, Y. Zhong, P. Lu, H. Peng, and J. Lenneman, "Monocular 3D vehicle detection using uncalibrated traffic cameras through homography," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Sep. 2021, pp. 3814–3821.
- [24] M. Chang, S. Moon, R. Mahjourian, and J. Kim, "BEVMap: Map-aware BEV modeling for 3D perception," in *Proc. IEEE/CVF Winter Conf. Appl. Comput. Vis. (WACV)*, Jan. 2024, pp. 7404–7413.
- [25] B. Zhou, J. Xie, Y. Pan, J. Wu, and C. Lu, "MotionBEV: Attention-aware online LiDAR moving object segmentation with bird's eye view based appearance and motion features," *IEEE Robot. Autom. Lett.*, vol. 8, no. 12, pp. 8074–8081, Dec. 2023.
- [26] S. Kedia, Y. Zhou, and S. H. Karumanchi, "Integrated perception and planning for autonomous vehicle navigation: An optimization-based approach," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW)*, vol. 1, Jun. 2023, pp. 3206–3215.
- [27] Z. Li, Z. Yu, W. Wang, A. Anandkumar, T. Lu, and J. M. Alvarez, "FB-BEV: BEV representation from forward-backward view transformations," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2023, pp. 6896–6905.
- [28] Z. Liu, S. Chen, X. Guo, X. Wang, T. Cheng, H. Zhu, Q. Zhang, W. Liu, and Y. Zhang, "Vision-based uneven BEV representation learning with polar rasterization and surface estimation," in *Proc. Conf. Robot Learn.*, 2023, pp. 437–446.
- [29] A. Das, S. Paul, N. Scholz, A. K. Malviya, G. Sistu, U. Bhattacharya, and C. Eising, "Fisheye camera and ultrasonic sensor fusion for near-field obstacle perception in bird's-eye-view," 2024, *arXiv:2402.00637*.
- [30] K. Dasgupta, A. Das, S. Das, U. Bhattacharya, and S. Yogamani, "Spatio-contextual deep network-based multimodal pedestrian detection for autonomous driving," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 9, pp. 15940–15950, Sep. 2022.

- [31] I. Barabanau, A. Artemov, E. Burnaev, and V. Murashkin, "Monocular 3D object detection via geometric reasoning on keypoints," in *Proc. 15th Int. Joint Conf. Comput. Vis., Imag. Comput. Graph. Theory Appl.*, 2020, pp. 652–659.
- [32] T. Wang, Z. H. U. Xinge, J. Pang, and D. Lin, "Probabilistic and geometric depth: Detecting objects in perspective," in *Proc. Conf. Robot Learn.*, 2022, pp. 1475–1485.
- [33] T. Wang, X. Zhu, J. Pang, and D. Lin, "FCOS3D: Fully convolutional one-stage monocular 3D object detection," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. Workshops (ICCVW)*, Oct. 2021, pp. 913–922.
- [34] W. Yang, Z. Li, C. Wang, and J. Li, "A multi-task faster R-CNN method for 3D vehicle detection based on a single image," *Appl. Soft Comput.*, vol. 95, Oct. 2020, Art. no. 106533.
- [35] N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, and S. Zagoruyko, "End-to-end object detection with transformers," in *Proc. Eur. Conf. Comput. Vis. Switzerland: Springer*, 2020, pp. 213–229.
- [36] Z. Li, W. Wang, H. Li, E. Xie, C. Sima, T. Lu, Y. Qiao, and J. Dai, "BEVFormer: Learning bird's-eye-view representation from multi-camera images via spatiotemporal transformers," in *Proc. Eur. Conf. Comput. Vis.*, S. Avidan, G. Brostow, M. Cissé, G. M. Farinella, and T. Hassner, Eds., Cham, Switzerland: Springer, 2022, pp. 1–18.
- [37] J. Philion and S. Fidler, "Lift, splat, shoot: Encoding images from arbitrary camera rigs by implicitly unprojecting to 3D," in *Proc. Eur. Conf. Comput. Vis.*, 2020, pp. 194–210.
- [38] P. Li, S. Ding, X. Chen, N. Hanselmann, M. Cordts, and J. Gall, "PowerBEV: A powerful yet lightweight framework for instance prediction in bird's-eye view," in *Proc. 32nd Int. Joint Conf. Artif. Intell.*, Aug. 2023, pp. 1080–1088.
- [39] S. A. Abbas and A. Zisserman, "A geometric approach to obtain a bird's eye view from an image," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. Workshop (ICCVW)*, Oct. 2019, pp. 4095–4104.
- [40] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun, "CARLA: An open urban driving simulator," in *Proc. 1st Annu. Conf. Robot Learn.*, 2017, pp. 1–16.
- [41] R. F. Benekohal and J. Treiterer, "CARSIM: Car-following model for simulation of traffic in normal and stop-and-go conditions," *Transp. Res. Rec.*, vol. 1194, pp. 99–111, Jan. 1988.
- [42] P. Sun et al., "Scalability in perception for autonomous driving: Waymo open dataset," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 2443–2451.
- [43] R. Xu, Z. Tu, H. Xiang, W. Shao, B. Zhou, and J. Ma, "CoBEVT: Cooperative bird's eye view semantic segmentation with sparse transformers," in *Proc. Conf. Robot Learn. (CoRL)*, 2022, pp. 989–1000.
- [44] E. Xie, Z. Yu, D. Zhou, J. Philion, A. Anandkumar, S. Fidler, P. Luo, and J. M. Alvarez, "M<sup>2</sup>BEV: Multi-camera joint 3D detection and segmentation with unified birds-eye view representation," 2022, *arXiv:2204.05088*.
- [45] Y. Zhang, Z. Zhu, W. Zheng, J. Huang, G. Huang, J. Zhou, and J. Lu, "BEVerse: Unified perception and prediction in birds-eye-view for vision-centric autonomous driving," 2022, *arXiv:2205.09743*.
- [46] S. Ettinger, S. Cheng, B. Caine, C. Liu, H. Zhao, S. Pradhan, Y. Chai, B. Sapp, C. Qi, Y. Zhou, Z. Yang, A. Chouard, P. Sun, J. Ngiam, V. Vasudevan, A. McCauley, J. Shlens, and D. Anguelov, "Large scale interactive motion forecasting for autonomous driving: The waymo open motion dataset," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2021, pp. 9690–9699.
- [47] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: The KITTI dataset," *Int. J. Robot. Res.*, vol. 32, no. 11, pp. 1231–1237, Sep. 2013.
- [48] B. Wilson, W. Qi, T. Agarwal, J. Lambert, J. Singh, S. Khandelwal, B. Pan, R. Kumar, A. Hartnett, J. K. Pontes, D. Ramanan, P. Carr, and J. Hays, "Argoverse 2: Next generation datasets for self-driving perception and forecasting," in *Proc. Neural Inf. Process. Syst. Track Datasets Benchmarks*, vol. 1, 2021.
- [49] J. Houston, G. Zuidhof, L. Bergamini, Y. Ye, L. Chen, A. Jain, S. Omari, V. Igloukov, and P. Ondruska, "One thousand and one hours: Self-driving motion prediction dataset," in *Proc. Conf. Robot Learn.*, Nov. 2021, pp. 409–418.
- [50] W. Zhan, L. Sun, D. Wang, H. Shi, A. Clause, M. Naumann, J. Kummerle, H. Konigshof, C. Stiller, A. de La Fortelle, and M. Tomizuka, "INTERACTION dataset: An INTERNATIONAL, adversarial and cooperative moTION dataset in interactive driving scenarios with semantic maps," 2019, *arXiv:1910.03088*.
- [51] X. Huang, P. Wang, X. Cheng, D. Zhou, Q. Geng, and R. Yang, "The ApolloScope open dataset for autonomous driving and its application," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 42, no. 10, pp. 2702–2719, Oct. 2020.
- [52] H. Fazlali, Y. Xu, Y. Ren, and B. Liu, "A versatile multi-view framework for LiDAR-based 3D object detection with guidance from panoptic segmentation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2022, pp. 17171–17180.
- [53] D. Xiao, M. Dianati, W. G. Geiger, and R. Woodman, "Review of graph-based hazardous event detection methods for autonomous driving systems," *IEEE Trans. Intell. Transp. Syst.*, vol. 24, no. 5, pp. 4697–4715, May 2023.
- [54] Q. Chen and X. Qi, "Residual graph convolutional network for bird's-eye-view semantic segmentation," in *Proc. IEEE/CVF Winter Conf. Appl. Comput. Vis.*, Oct. 2024, pp. 3324–3331.
- [55] J. Fang, F. Wang, J. Xue, and T.-S. Chua, "Behavioral intention prediction in driving scenes: A survey," *IEEE Trans. Intell. Transp. Syst.*, vol. 25, no. 8, pp. 8334–8355, Aug. 2024.
- [56] S. Yao, R. Guan, X. Huang, Z. Li, X. Sha, Y. Yue, E. G. Lim, H. Seo, K. L. Man, X. Zhu, and Y. Yue, "Radar-camera fusion for object detection and semantic segmentation in autonomous driving: A comprehensive review," *IEEE Trans. Intell. Vehicles*, vol. 9, no. 1, pp. 2094–2128, Jan. 2024.
- [57] Y. Zhou, H. Zheng, X. Huang, S. Hao, D. Li, and J. Zhao, "Graph neural networks: Taxonomy, advances, and trends," *ACM Trans. Intell. Syst. Technol.*, vol. 13, no. 1, pp. 1–54, Feb. 2022.
- [58] S. Casas, C. Gulino, R. Liao, and R. Urtasun, "SpAGNN: Spatially-aware graph neural networks for relational behavior forecasting from sensor data," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2020, pp. 9491–9497.
- [59] S. Shiwakoti, S. B. Shahi, and P. Singh, "Deep learning methods for vehicle trajectory prediction: A survey," in *Proc. Int. Conf. IoT Based Control Netw. Intell. Syst. Cham, Switzerland: Springer*, 2023, pp. 539–554.
- [60] C. Nie, Z. Ju, Z. Sun, and H. Zhang, "3D object detection and tracking based on LiDAR-camera fusion and IMM-UKF algorithm towards highway driving," *IEEE Trans. Emerg. Topics Comput. Intell.*, vol. 7, no. 4, pp. 1242–1252, Jun. 2023.
- [61] J. Beck, R. Arvin, S. Lee, A. Khattak, and S. Chakraborty, "Automated vehicle data pipeline for accident reconstruction: New insights from LiDAR, camera, and radar data," *Accident Anal. Prevention*, vol. 180, Feb. 2023, Art. no. 106923.
- [62] A. M. Anisha, M. Abdel-Aty, A. Abdelraouf, Z. Islam, and O. Zheng, "Automated vehicle to vehicle conflict analysis at signalized intersections by camera and LiDAR sensor fusion," *Transp. Res. Rec.*, vol. 2677, no. 5, pp. 117–132, 2023.
- [63] T. Phan-Minh, E. C. Grigore, F. A. Boulton, O. Beijbom, and E. M. Wolff, "CoverNet: Multimodal behavior prediction using trajectory sets," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 14062–14071.
- [64] N. Deo, E. Wolff, and O. Beijbom, "Multimodal trajectory prediction conditioned on lane-graph traversals," in *Proc. 5th Conf. Robot Learn.*, Jan. 2022, pp. 203–212.
- [65] H. Cheng, M. Liu, L. Chen, H. Broszio, M. Sester, and M. Y. Yang, "GATraj: A graph- and attention-based multi-agent trajectory prediction model," *ISPRS J. Photogramm. Remote Sens.*, vol. 205, pp. 163–175, Nov. 2023.
- [66] M. Lee, S. S. Sohn, S. Moon, S. Yoon, M. Kapadia, and V. Pavlovic, "MUSE-VAE: Multi-scale VAE for environment-aware long term trajectory prediction," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2022, pp. 2211–2220.
- [67] L. Zhang, P. H. Su, J. Hoang, G. C. Haynes, and M. Marchetti-Bowick, "Map-adaptive goal-based trajectory prediction," in *Proc. Conf. Robot Learn.*, vol. 155, J. Kober, F. Ramos, and C. Tomlin, Eds., 2020, pp. 1371–1383.
- [68] M. Shah, Z. Huang, A. Laddha, M. Langford, B. Barber, S. Zhang, C. Vallespi-Gonzalez, and R. Urtasun, "LiRaNet: End-to-end trajectory prediction using spatio-temporal radar fusion," in *Proc. Conf. Robot Learn.*, 2020, pp. 31–48.
- [69] S. Fadadu, S. Pandey, D. Hegde, Y. Shi, F.-C. Chou, N. Djuric, and C. Vallespi-Gonzalez, "Multi-view fusion of sensor data for improved perception and prediction in autonomous driving," in *Proc. IEEE/CVF Winter Conf. Appl. Comput. Vis. (WACV)*, Jan. 2022, pp. 3292–3300.

- [70] Y. Sun, W. Zuo, and M. Liu, "See the future: A semantic segmentation network predicting ego-vehicle trajectory with a single monocular camera," *IEEE Robot. Autom. Lett.*, vol. 5, no. 2, pp. 3066–3073, Apr. 2020.
- [71] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 30, I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds., 2017, pp. 1–11.
- [72] Z. Sheng, Y. Xu, S. Xue, and D. Li, "Graph-based spatial-temporal convolutional network for vehicle trajectory prediction in autonomous driving," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 10, pp. 17654–17665, Oct. 2022.
- [73] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," in *Proc. Int. Conf. Learn. Represent.*, 2017.
- [74] X. Li and J. Saúde, "Explain graph neural networks to understand weighted graph features in node classification," in *Machine Learning and Knowledge Extraction*. Cham, Switzerland: Springer, 2020, pp. 57–76. [Online]. Available: [https://link.springer.com/chapter/10.1007/978-3-030-57321-8\\_4#citeas](https://link.springer.com/chapter/10.1007/978-3-030-57321-8_4#citeas)
- [75] S. Cantürk, R. Liu, O. Lapointe-Gagné, V. Létourneau, G. Wolf, D. Beaini, and L. Rampásek, "Graph positional and structural encoder," in *Proc. Int. Conf. Mach. Learn.*, 2024, pp. 5533–5566.
- [76] Y. Chai, B. Sapp, M. Bansal, and D. Anguelov, "MultiPath: Multiple probabilistic anchor trajectory hypotheses for behavior prediction," in *Proc. Conf. Robot Learn.*, 2020, pp. 86–99.
- [77] K. Messaoud, N. Deo, M. M. Trivedi, and F. Nashashibi, "Trajectory prediction for autonomous driving based on multi-head attention with joint agent-map representation," in *Proc. IEEE Intell. Vehicles Symp. (IV)*, Jul. 2021, pp. 165–170.
- [78] G. Elghazaly, R. Frank, S. Harvey, and S. Safko, "High-definition maps: Comprehensive survey, challenges, and future perspectives," *IEEE Open J. Intell. Transp. Syst.*, vol. 4, pp. 527–550, 2023.
- [79] R. Liu, J. Wang, and B. Zhang, "High definition map for automated driving: Overview and analysis," *J. Navigat.*, vol. 73, no. 2, pp. 324–341, Mar. 2020.
- [80] A. Chalvatzaras, I. Pratikakis, and A. A. Amanatiadis, "A survey on map-based localization techniques for autonomous vehicles," *IEEE Trans. Intell. Vehicles*, vol. 8, no. 2, pp. 1574–1596, Feb. 2023.
- [81] T. Salzmann, B. Ivanovic, P. Chakravarty, and M. Pavone, "Trajectron++: Dynamically-feasible trajectory forecasting with heterogeneous data," in *Proc. Eur. Conf. Comput. Vis.* Glasgow, U.K. Cham, Switzerland: Springer, Aug. 2020, pp. 683–700.
- [82] N. Djuric, V. Radosavljevic, H. Cui, T. Nguyen, F.-C. Chou, T.-H. Lin, N. Singh, and J. Schneider, "Uncertainty-aware short-term motion prediction of traffic actors for autonomous driving," in *Proc. IEEE Winter Conf. Appl. Comput. Vis. (WACV)*, Mar. 2020, pp. 2084–2093.
- [83] M. Tan and Q. V. Le, "EfficientNet: Rethinking model scaling for convolutional neural networks," 2019, *arXiv:1905.11946*.
- [84] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 2999–3007.
- [85] Anqi Mao, Mehryar Mohri, and Yutao Zhong, "Cross-entropy loss functions: Theoretical analysis and applications," in *Proc. Int. Conf. Mach. Learn.*, 2023, pp. 23803–23828.
- [86] I. Loshchilov and F. Hutter, "Decoupled weight decay regularization," in *Proc. 7th Int. Conf. Learn. Represent.*, New Orleans, LA, USA, May 2019, pp. 1–12.
- [87] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014, *arXiv:1412.6980*.
- [88] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer, "Automatic differentiation in PyTorch," in *Proc. NIPS Autodiff Workshop, Future Gradient-based Mach. Learn. Softw. Techn.*, Dec. 2017.
- [89] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby, "An image is worth 16×16 words: Transformers for image recognition at scale," in *Proc. Int. Conf. Learn. Represent.*, 2021.
- [90] N. Gosala and A. Valada, "Bird's-eye-view panoptic segmentation using monocular frontal view images," *IEEE Robot. Autom. Lett.*, vol. 7, no. 2, pp. 1968–1975, Apr. 2022.
- [91] S. Gong, X. Ye, X. Tan, J. Wang, E. Ding, Y. Zhou, and X. Bai, "GitNet: Geometric prior-based transformation for birds-eye-view segmentation," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*. Cham, Switzerland: Springer, 2022, pp. 396–411.



**SUSHIL SHARMA** received the B.Tech. degree in electronics and instrumentation from ITM University, India, in 2014, and the M.Sc. degree in automatic control and robotics, specializing in smart aerospace and autonomous systems, from Université Paris-Saclay, France, and Poznań University of Technology, Poland, in 2017. He is currently pursuing the Ph.D. degree with the Department of Electronic and Computer Engineering, University of Limerick, Ireland. His research interests include "automotive parking systems" involve a broad range of areas, such as multicamera perception, trajectory prediction, and machine-based algorithms.



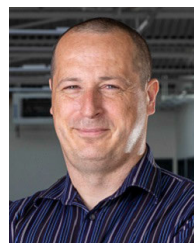
**ARINDAM DAS** is currently pursuing the Ph.D. degree with the Department of Electronic and Computer Engineering, University of Limerick, Ireland. He is also an AI Software Architect with the Department of Driving Software and Systems (DSW), Valeo, India, where he holds the position of an Expert in AI. He is responsible for designing AI algorithms to support various features for autonomous driving systems. He has ten years of industry experience in computer vision, deep learning, and document analysis. He has authored 17 peer-reviewed publications and 52 patents. His current research interests include weakly-supervised learning, domain adaptation, image restoration, and multimodal learning.



**GANESH SISTU** is currently a Principal AI Architect at Valeo, Ireland, where he leads global teams in applied research and product development, focusing on automated driving and parking perception. With a solid background of over 13 years in computer vision and machine learning, he has made notable contributions to the field, authoring over 30 publications in respected conferences, including ICCV, WACV, ICRA, and GECCO. Balancing his industrial role, he also engages in academia as an Adjunct Assistant Professor with the University of Limerick, Ireland. His commitment extends to guiding emerging talents in AI, serving on the industry board for the Science Foundation Ireland Data Science Ph.D. studies and the University of Limerick's National M.Sc. studies in AI Programs.



**MARK HALTON** (Member, IEEE) received the B.A. and B.A.I. degrees in mechanical engineering and mathematics from Trinity College Dublin, Ireland, and the M.Eng. and Ph.D. degrees from the University of Limerick, Ireland. From 2007 to 2022, he was a Senior Research Fellow with the University of Limerick, and in 2022, he took up a position as an Associate Professor with the Electronic and Computer Engineering Department. To date, he has authored and co-authored over 50 peer-reviewed conference and journal publications. His current research interests include control, AI, and automotive.



**CIARÁN EISING** (Senior Member, IEEE) received the Ph.D. degree from NUI Galway, in 2010. From 2009 to 2020, he was the Computer Vision Team Lead at Valeo Vision Systems, where he also held the title of a Senior Expert. In 2020, he joined the University of Limerick, Ireland, as an Associate Professor of computer engineering.

...