Road User Specific Trajectory Prediction in Mixed Traffic Using Map Data

Boekema, Hidde J.H.; Moustafa, Emran Yasser ; Kooij, Julian F.P.; Gavrila, Dariu M.

DOI
10.1109/LRA.2025.3564746

Publication date
2025

Document Version
Final published version

Published in
IEEE Robotics and Automation Letters

Road User Specific Trajectory Prediction in Mixed Traffic Using Map Data

Hidde J-H. Boekema, *Graduate Student Member, IEEE*, Emran Yasser Moustafa, Julian F.P. Kooij, and Dariu M. Gavrila, *Member, IEEE*

***Abstract*—This paper studies road user trajectory prediction in mixed traffic, i.e. where vehicles and Vulnerable Road Users (VRUs, i.e. pedestrians, cyclists and other riders) closely share a common road space. We investigate if typical prediction components (scene graph representation, scene encoding, waypoint prediction, motion dynamics) should be specific to each road user class. Using the recent VRU-heavy View-of-Delft Prediction (VoD-P) dataset, we study several directions to improve the performance of the state-of-the-art map-based prediction models (PGP, TNT) in urban settings. First, we consider the use of class-specific map representations. Second, we investigate if the weights of different components of the model should be shared or separated by class. Finally, we augment VoD-P training data with automatically extracted trajectories from the 360-degree LiDAR scans by the recording vehicle. This data is made publicly available. We find that pre-training the model on auto-labels and making it class-specific leads to a reduction of up to 22.2%, 20.0%, and 18.2% in minADE ($K = 10$ samples) for pedestrians, cyclists, and vehicles, respectively.**

***Index Terms*—Deep learning methods, intention recognition.**

I. Introduction

AUTOMATED vehicles need to predict the trajectories of surrounding agents to ensure safe and comfortable driving behaviour. This is especially true for mixed traffic scenarios, where vehicles and Vulnerable Road Users (VRUs, e.g. pedestrians and cyclists) closely share a common road space. Trajectory prediction is challenging in such scenarios due to the close proximity of agents, their manoeuvrability, and complex interactions.

Semantic map data helps better predict agents' trajectories in challenging scenarios [1], [2], [3], [4]. However, most high-quality urban trajectory datasets that provide semantic map data [5], [6], [7] lack interactions between vehicles and VRUs. For instance, large-scale datasets such as nuScenes [5], Argoverse 2 [7], and Waymo Motion [6] were recorded in urban areas with a high degree of segregation between road user classes. This gap was recently addressed by the View-of-Delft Prediction [8] (VoD-P) dataset, built on the multi-sensor View-of-Delft [9] (VoD) object detection dataset recorded in the historical centre of Delft. VoD-P contains annotated tracks in mixed traffic scenarios, multi-modal sensor data (camera, radar, and 360° LiDAR), and a semantic map with lane, sidewalk, and crosswalk annotations.

A previous evaluation of state-of-the-art trajectory prediction models, which were primarily developed on vehicle-heavy datasets, revealed that *Prediction via Graph-based Policy* (PGP) [2] was best-performing on this VoD-P benchmark[8]. The evaluation also revealed that PGP performed comparatively poorly for VRUs. In this paper, we study several potential reasons and propose solutions to make map-based models, such as PGP and the well-cited TNT [10], better suited to mixed traffic scenarios.

First, PGP uses map information to help predict trajectories for all agents, but typical scene graph representations only represent roads for vehicles. However, cyclists are not as bound to the lanes and driving directions as vehicles are, and pedestrians primarily traverse areas (e.g. sidewalks, squares) outside of the lanes. Therefore, we argue that vehicle-centric lane graphs do not serve all agent classes and propose to extend the lane graph with agent-specific nodes and edges.

Second, PGP uses a shared network architecture for all target agents. Although the class of the target agent is encoded in the scene representation and hence, in principle, accessible at later stages in the network, we argue that prediction models like PGP could benefit from specialising certain components for specific agent classes. In this paper, we study which generic components (scene encoding, waypoint prediction, and motion/trajectory prediction) should and should not be conditioned on the target agent class.

Third, since the trajectories in the VoD-P dataset were based on VoD objects annotated in the view of the front-facing camera [9], the field of view of the camera limited the number of trajectories. Therefore, we propose extending VoD-P with additional trajectories by automatically detecting and tracking all surrounding road users from the 360° LiDAR scans. Although such trajectories are of lower quality than those based on manual annotations, this can still help improve prediction for all classes.

II. Related Work

Many approaches have been proposed for predicting the trajectories of traffic agents; see [11], [12], [13], [14], [15] for

Received 31 October 2024; accepted 12 April 2025. Date of publication 28 April 2025; date of current version 8 May 2025. This article was recommended for publication by Associate Editor S. Thakar and Editor A. Banerjee upon evaluation of the reviewers' comments. This work was supported by Dutch Research Council (NWO), through Efficient Deep Learning (EDL) Project under Project P16-25. *(Corresponding author: Dariu M. Gavrila.)*

The authors are with Intelligent Vehicles Group, TU Delft, 2628 CD Delft, The Netherlands (e-mail: d.m.gavrila@tudelft.nl).

Data will be made freely available for non-profit entities.

This article has supplementary downloadable material available at https://doi.org/10.1109/LRA.2025.3564746, provided by the authors.

Digital Object Identifier 10.1109/LRA.2025.3564746

surveys. With the advent of high-quality semantic map data in datasets such as nuScenes [5], Argoverse 2 [7], Waymo Motion [6], and View-of-Delft Prediction [8], many recent approaches use map-based contextual cues to inform predictions [1], [2], [3], [4], [10], [16], [17], [18], [19], [20], [21], [22]. We briefly discuss these approaches here.

Vehicles are more constrained by the road layout than VRUs. Hence, vehicle-centric map-based trajectory prediction approaches primarily use road features in their map representation. These approaches can be categorised according to whether they use map data implicitly (as context) or explicitly (for intent estimation).

Approaches that use map data implicitly fuse this data with other input modalities to improve predictions with respect to the static and, in some cases, the dynamic environment. For example, reasoning about the future interactions of vehicles with a lane graph is done in [18] using graph convolutions and in [23] using attention. Attention is also used in Wayformer [24] to fuse different input modalities (including map data) in an efficient way. MTR [1] collects map elements along predicted trajectories to refine the predictions and improve their consistency with the static environment. SEPT [4] performs agent-lane attention on a sub-graph created by pruning the full road graph using a path planner to identify the most relevant roads for the target agent [25].

A common paradigm of explicit map-based approaches is estimating coarse goals based on motion states and map data (e.g. traversals over lanes) and refining these into trajectories. For example, TNT [10] estimates possible target locations from sample poses along lane centrelines and regresses trajectories towards these targets; [16] modifies this by sampling goal locations on the entire drivable area. These approaches encode the lane graph using the graph neural network-based VectorNet [21] model; another popular lane graph encoder is LaneGCN [22]. PGP [2] instead creates a directed graph of lane centrelines and models agent intent using a traversal probability along the graph edges. GOHOME [17] similarly estimates coarse intention using a graph representation but additionally uses these estimates to generate a fine-grained trajectory heatmap. Similarly, SemanticFormer [3] creates "meta-paths" from a lane (knowledge) graph that represent feasible road traversals for agents. Instead of explicitly conditioning predictions on the lanes, P2T [19] predicts goal and path states from a raster image of the map. However, these approaches share the environment representation and model between all road user classes despite different information being relevant for different classes.

A. Contributions

Compared to the related work, our contributions are:

1) We propose to use class-specific map representations to improve trajectory prediction models, in particular for VRUs (Section III-B). These class-specific representation contain features relevant for estimating the intent of each road user class, e.g. a lane-oriented star graph for pedestrians. We test our class-specific representations using the state-of-the-art PGP [2] model and the popular TNT [10] model. This leads to better predictions across all three classes on the View-of-Delft Prediction [8] (VoD-P) dataset.
2) We investigate which model components of the PGP model should be class-specific (Section III-C). We show that using class-specific encoders and decoders (with class-specific graphs) leads to the best performance across all classes in the VoD-P dataset.
3) We augment the data in the VoD-P dataset using labels from an auto-labelling pipeline we developed, and show that pre-training our model with the augmented data leads to improved predictions on the VoD-P test set (Section III-D). The augmented data is released as a pre-training dataset for VoD-P.

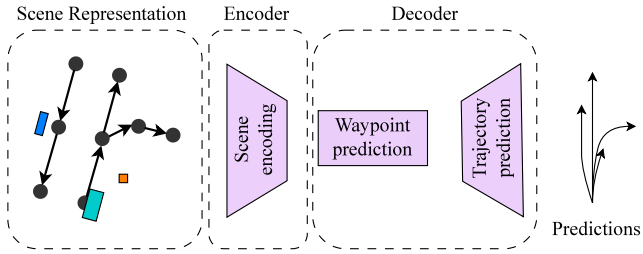


Fig. 1. Generic model components for map-based prediction models. We investigate the prediction performance when using a class-specific scene representation, encoder, and decoder.

III. Proposed Method

Vector map-based trajectory prediction models are among the most successful approaches on various benchmarks [1], [2], [3], [4]. We select the graph-based PGP [2] model as the baseline for our experiments, as this model is one of the best-performing models on the nuScenes [5] and VoD-P [8] datasets. Furthermore, PGP consists of several components which perform subtasks also found in similar models [1], [3], [10], [16], [17], [19], and therefore well-suited to study the impact of making such components class-specific; see Fig. 1 for an overview of these model components.

In Section III-A, we provide an overview of PGP and its components. Section III-B introduces the various ways we consider to extend the scene representation. Then, Section III-C details how we built variants of the PGP model. Finally, Section III-D describes our procedure for extending VoD-P with auto-labelled trajectories.

A. Overview of PGP Model

The input of PGP is a graph representation of the scene, containing lane information from the semantic map, temporal data from the ego-agent, and all detected nearby agents. One agent in the scene is selected as prediction target, where this agent can be a pedestrian, cyclist or vehicle. We here present an overview of the scene representation, and the main components of the baseline PGP [2]. Later subsections will then propose variants of both.

1) Scene Representation: The scene is represented as a heterogeneous directed graph $G = (V, E)$, where V is the set of nodes and $E \subseteq V \times V$ the edges between them. This graph captures the possible motions of an agent at a particular location.

Nodes represent the geometry of map elements. We follow [2] in encoding lane centreline segments as nodes to ensure that nodes represent geometries of similar length. Specifically, lane centrelines are subdivided into equal-length segments of N poses such that each node $v \in V$ consists of a sequence of vectors $f^v_{1:N} = [f^v_1, \ldots, f^v_N]$. Each $f^v_n = [x^v_n, y^v_n, \theta^v_n, I^v_n]$, where x^v_n, y^v_n, and θ^v_n are the position and yaw of the n^{th} pose of v and I^v_n is a 2-D binary vector indicating whether the pose lies on a stop line or crosswalk.

Edges define the valid or legal connections between map elements. There are two types of edges in the graph: successor and proximal edges. Successor edges indicate the legal connections between lane centreline segments. Proximal edges connect nodes that lie within a certain distance of each other. The edges are labelled according to their type, allowing the prediction model to distinguish between them.

2) PGP Model Structure: PGP consists of an encoder stage and a decoder stage. Within these stages we can identify several components that are typical for many trajectory prediction models, namely: scene representation, scene encoding, waypoint prediction, and trajectory prediction.

PGP Encoder: The encoder is a graph neural network with weights θ^{E}, which extracts features from the *scene representation* $G = (V, E)$ and past motion states $X = \{\mathbf{x}^a \mid a \in A\}$, where A is the set of agents in the scene and $\mathbf{x}^a$ contains the motion state history of agent a,

$$S = \text{Enc}(V, E, X; \theta^{\text{E}}). \tag{1}$$

where S is the resulting scene encoding, and $\text{Enc}(\cdot)$ is the "Graph Encoder" network of [2]. The *scene encoding* S assigns to nodes in V a latent scene feature containing information about the relevant traffic context at a node. This is done by fusing the motion state history of agents in X in close proximity to a given node $v \in V$ with the features of v using attention. Note that we follow [8] in adding a class label for the target agent to the motion states.

PGP Decoder: The decoder stage uses the scene encoding S to generate K possible future trajectories $Y = \{\mathbf{y}^{a_0}_k \mid k \in \{1, \ldots, K\}\}$ for a target agent a_0, where $\mathbf{y}^{a_0}$ are the predicted future motion states of a_0.

PGP does this by first performing *waypoint prediction*, i.e. estimating (intermediate) goal locations that represent high-level decisions for the agent (e.g. turn left or right, stay or move along road, etc.). A "Policy Header" network [2] with parameters θ^{P} estimates a "policy" P for the target agent from the scene encoding. The policy assigns transition probabilities over the edges E. Sampling node transitions from this policy yields a node sequence $T = [v_1, v_2, \ldots, v_M]$ that represent waypoints on the scene representation (a process also referred to as "policy roll-out"). This sampling of the policy can be repeated K times,

$$P = \text{Policy}(E, S; \theta^{\text{P}}), \tag{2}$$

$$T_k = \text{sample}(P, V), \qquad \text{for } 1 \leq k \leq K. \tag{3}$$

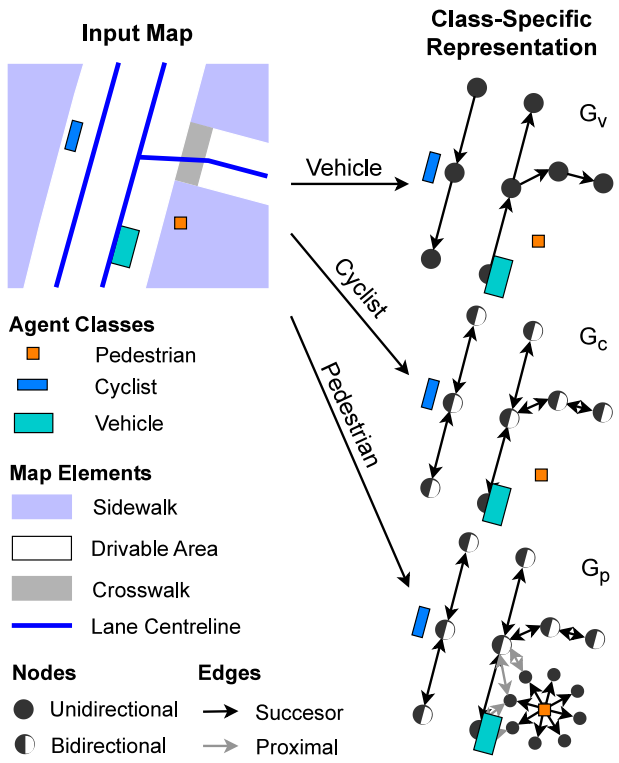


Fig. 2. Overview of our class-specific scene representations. We extract lane graphs with features specific to the target agent class from a semantic map. For vehicles, graph G_{v} is a lane graph with unidirectional (legal) edges. The cyclist graph G_{c} has bidirectional nodes and edges. Pedestrian intent is captured by a lane-oriented star graph around the pedestrian, which is connected to G_{c} to share social context in G_{p}; see Fig. 3 for the lane-oriented reference frame.

Finally, for each waypoint sequence T_k, the "Trajectory Decoder" [2] regresses a smooth future trajectory that should consider the agent's motion dynamics to get a *trajectory prediction:*

$$\mathbf{y}^{a_0}_k = \text{Traj}(T_k, \mathbf{z}_k, \mathbf{m}^{a_0}; \theta^{\text{T}}). \tag{4}$$

Here $\mathbf{m}^{a_0} = \text{GRU}(\mathbf{x}^{a_0})$ encodes the target agent's past motion, and $\mathbf{z}_k$ is a continuous random variable used to generate local motion variations in the generated trajectories even for similar node sequences T. The decoder $\text{Traj}(\cdot)$ is an MLP and is therefore not constrained to following T, which is useful for scenarios where agents do not follow the lane graph e.g. a vehicle reversing in a one-way street.

B. Class-Specific Scene Representations

PGP's default graph-based scene representation is vehicle-centric; it represents road lane information and it forces policies to adhere to the legal driving direction. Since VRUs are not restricted to the same road rules as vehicles in city centres like in VoD-P [8], we extend this representation to class-specific instances for vehicles, cyclists and pedestrians. See Fig. 2 for an overview.

Vehicles: Only nodes V_{v} representing lane centreline segments and edges E_{v} for legal transitions between them are included in

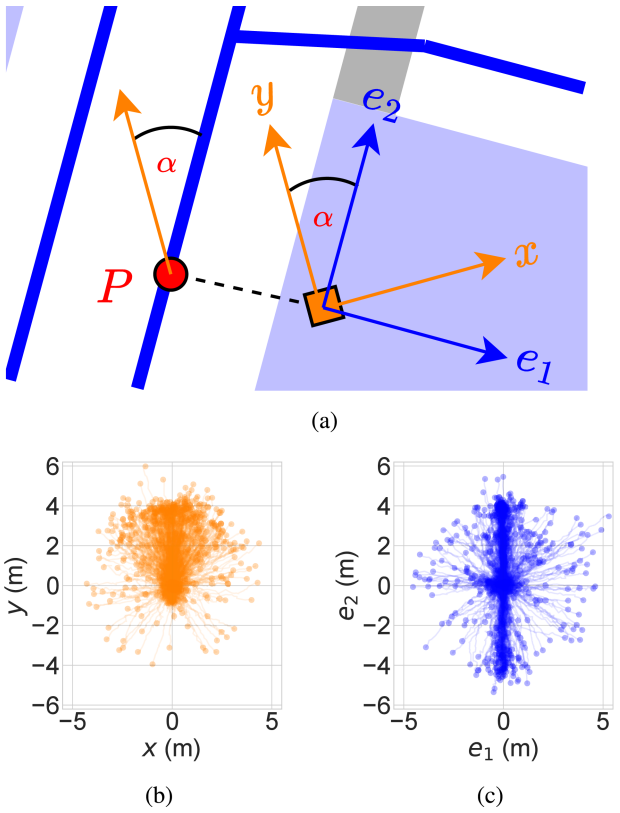


Fig. 3. (a) Agent-centric (x, y) and lane-centric (e_1, e_2) reference frames. The agent-centric frame is rotated by the orientation α of the lane point P closest to the pedestrian (orange square) to obtain the lane-centric frame. The pedestrian trajectories in the VoD-P training set are plotted in the (b) agent-centric and (c) lane-centric reference frames. Pedestrians mostly move along lanes in VoD-P.

the vehicle-specific representation G_{v}, as vehicles such as cars and trucks are typically limited to driving on the lanes of the road, and must strictly follow the legally-designated driving direction of the lanes.

Cyclists: Like vehicles, cyclists are mostly constrained to specially designated areas such as bike lanes. However, in urban settings they may also share the road with vehicles, and are often not required to (or simply do not) follow the direction of the lane. Hence, we invert the geometries of the original nodes V_{v} to obtain "inverted" nodes V'_{v}, and compose the set of cyclist nodes $V_{\mathrm{c}} = \{V_{\mathrm{v}}, V'_{\mathrm{v}}\}$ from these sets. An edge is created between the nodes in V_{c} if their lane segments are adjacent (i.e. one segment ends and another ends at the same location). This yields the cyclist graph $G_{\mathrm{c}} = (V_{\mathrm{c}}, E_{\mathrm{c}})$ with $V_{\mathrm{c}} \supset V_{\mathrm{v}}$ and $E_{\mathrm{c}} \supset E_{\mathrm{v}}$.

Pedestrians: This road user class is the least constrained in its freedom of movement in urban settings. Fig. 3 shows a histogram of the (future) endpoints of pedestrian trajectories from the VoD-P [8] dataset, where the trajectories were oriented with respect to the nearest lane to the agent at the last observed time step. Most trajectories follow the lane (represented by the e_2-axis), or are stationary. A minority move orthogonal to the lane, signifying crossing behaviour. Hence the lane features are less relevant to pedestrians than their orientation (and position) with respect to the lane. Since the graph captures the high-level intent of the target agent, we create a star graph with k_{S} leaf nodes centred at the position of the target agent and oriented with the nearest lane. The leaf nodes contain line segments with orientation equal to the node angle and length corresponding to the average pedestrian velocity in the training set; the internal node contains a point (to capture the stationary mode observed in Fig. 3). To share information from the surrounding lanes (e.g. if other agents are approaching the pedestrian), we connect the star graph to the cyclist scene representation G_{c}. We do this by adding bidirectional proximal edges between nodes of the two graphs if they lie within a distance threshold d_{prox} of each other. This gives the pedestrian scene representation $G_{\mathrm{p}} = (V_{\mathrm{p}}, E_{\mathrm{p}})$ with $V_{\mathrm{p}} \supset V_{\mathrm{c}}$ and $E_{\mathrm{p}} \supset E_{\mathrm{c}}$.

C. Class-Specific Model Components

The baseline PGP model described in Section III-A uses the same network weights irrespective of the target agent class. In this section, we describe various potential modifications of this baseline to investigate which components can be shared, and which benefit from becoming class-specific.

- *PGP [2]. shared encoder-decoder, shared scene representation:* The original PGP model, which shares the encoding and decoding stages through the same network parameters $\theta^{\mathrm{E}}, \theta^{\mathrm{P}}, \theta^{\mathrm{T}}$ for all target classes, and uses the vehicle-centric scene representation G_{v} for all classes.
- *PGP-D. class-specific decoder, shared scene representation:* The scene encoding is still shared between classes, and the scene representation is still G_{v}. However, this variant applies separate decoders for each agent class $c \in \{\mathrm{v}, \mathrm{c}, \mathrm{p}\}$ using class-specific network parameters θ^{P}_c and θ^{T}_c.

$$S = \mathrm{Enc}(V_{\mathrm{v}}, E_{\mathrm{v}}, X; \theta^{\mathrm{E}}), \tag{5}$$

$$P = \mathrm{Policy}(E_{\mathrm{v}}, S; \theta^{\mathrm{P}}), \tag{6}$$

$$\mathbf{y}_k^{a_0} = \mathrm{Traj}(T_k, \mathbf{z}_k, \mathbf{m}^{a_0}; \theta^{\mathrm{T}}_c). \tag{7}$$

- *PGP-ED. class-specific encoder-decoder, shared scene representation:* This variant uses *fully separate networks* (i.e. class-specific encoding and decoding parameters θ^{E}_c, θ^{P}_c and θ^{T}_c) for each target class c. All classes still use G_v as the encoder input and for their policy's waypoint generation.

$$S = \mathrm{Enc}(V_{\mathrm{v}}, E_{\mathrm{v}}, X; \theta^{\mathrm{E}}_c), \tag{8}$$

$$P = \mathrm{Policy}(E_{\mathrm{v}}, S; \theta^{\mathrm{P}}_c), \tag{9}$$

$$\mathbf{y}_k^{a_0} = \mathrm{Traj}(T_k, \mathbf{z}_k, \mathbf{m}^{a_0}; \theta^{\mathrm{T}}_c). \tag{10}$$

- *PGP-EDG. class-specific encoder-decoder, class-specific scene representation:* The final variant uses fully separate network parameters for each class, similar to PGP-ED, but uses the class-specific scene representations. The policy only selects waypoints on the nodes V_c and edges E_c available to target class $c \in \{\mathrm{v}, \mathrm{c}, \mathrm{p}\}$:

$$S = \mathrm{Enc}(V_c, E_c, X; \theta^{\mathrm{E}}_c), \tag{11}$$

$$P = \mathrm{Policy}(E_c, S; \theta^{\mathrm{P}}_c), \tag{12}$$

TABLE I
NOMENCLATURE OF MODEL PERMUTATIONS

Name	Separate Encoder	Separate Decoder	Separate Scene Representation
PGP [2]	☐	☐	☐
PGP-D	☐	☑	☐
PGP-ED	☑	☑	☐
PGP-EDG	☑	☑	☑

$$\mathbf{y}_k^{a_0} = \text{Traj}(T_k, \mathbf{z}_k, \mathbf{m}^{a_0}; \theta_c^{\text{T}}). \tag{13}$$

These variants are summarised in Table I. Note that we do not try a class-specific encoder with shared decoder and graphs (i.e. PGP-E) because the decoder would need to able to make predictions from three distinct feature spaces. Similarly, it is illogical to have class-specific scene representations without a class-specific encoder.

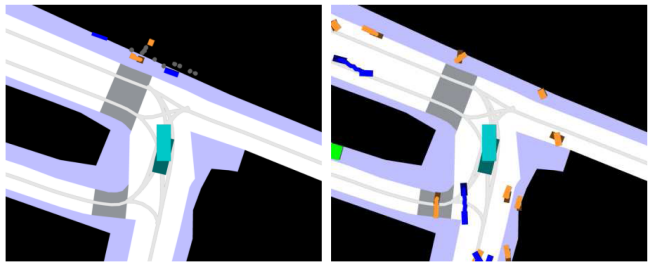
D. Data Augmentation Using Auto-Labelling (AL)

The View-of-Delft Prediction dataset only contains high-quality agent trajectories from manually annotated bounding boxes in the front-facing camera view (60° FoV) of the recording vehicle. This affects the quantity and quality of the training data, as there are also many agents outside of this annotated region. Since the scans from the roof-mounted LiDAR provides a 360° view around the vehicle, we create an auto-labelling (AL) pipeline to automatically detect and track agents in these pointclouds.

First, we train PointPillars [26] using the front-facing manually annotated bounding boxes. Afterwards, we input rotated slices of the pointcloud to the trained detector to generate detections in the unnannotated regions. The detections are stitched together with non-maximal suppression to obtain comprehensive 360° detections. Next, we apply the Immortal Tracker [27] to generate tracks from these detections.

Our AL pipeline produces labels for the three main agent classes: pedestrians, cyclists, and cars. Only agents observed at least five times are kept in the final prediction set, where an observation refers to the association of a detection to a specific track ID within a single frame. Optionally, a human annotator can manually match and prune the tracks to enhance prediction accuracy. Pruned tracks are excluded from the set of tracked objects. Linear interpolation is used to estimate the unobserved states of the retained tracks.

The AL tracks undergo the same filtering steps as the annotated tracks of the VoD-P dataset; refer to [8] for more details. The AL dataset retains a significantly higher number of dynamic target agents than the manually annotated dataset after this filtering step; see Fig. 4 for an example. As detailed in Table II, the number of valid tracked target agents has substantially increased: 91.1% for vehicles, 182.8% for cyclists, and 121.9% for pedestrians. The number of unique tracked agents has increased from 353 to 587. Fig. 5 shows that the distribution of the velocities of target agents is similar for the auto-labelled and annotated train sets, with slightly more static pedestrians and vehicles in the AL set.



(a) Annotated (b) Auto-Labelled

Fig. 4. Example scenario from VoD-P train set, shown with annotations and auto-labels.

TABLE II
NUMBER OF TARGET AGENTS IN THE MANUALLY ANNOTATED AND AUTO-LABELLED VOD-P TRAINING SETS

Training Set	Number of Agents			
	Vehicles	Vehicles (excl. ego)	Cyclists	Pedestrians
Annotated	436	110	261	699
Auto-Labelled	833	508	738	1551

IV. EXPERIMENTS

In this section, we discuss the experimental setup used to evaluate our approach and the results of the experiments.

A. Setup

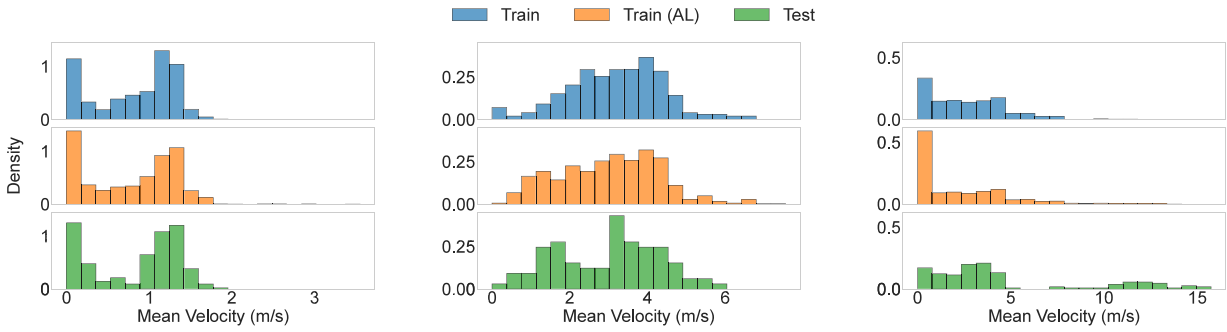
We evaluate our approach on the VRU-heavy VoD-P [8] dataset. However, the VoD-P test set has a scenario with vehicles on a residential road driving at velocities >10 m/s, as Fig. 5(c) shows. Fig. 5(c) also shows that there are few vehicles equally fast in the train set, making this scenario challenging to predict for. Since this paper focuses on urban scenarios involving interaction between different road user classes, we split the test set vehicles into "Vehicle (Low-Speed)" (≤ 10 m/s) and "Vehicle (High-Speed)" (>10 m/s) subsets. We report our results on both subsets.

To show the generalisability of our approach beyond the PGP [2] model, we additionally evaluate it on the well-known TNT [10] model. We substitute the original VectorNet backbone of the TNT model by the PGP encoder to test our scene representation, but maintain the original components of the rest model. We will refer to this model as TNT*.

To assess the models, we use common metrics (e.g. [8]):

- *minADE*$_K$*:* The minimum average displacement error over the K most likely predictions.
- *MissRate*$_{K,R}$*:* The fraction of samples for which the minimum final displacement error out of the K most likely predictions is farther than distance R from the ground truth position.

We select $K = 10$ and $R = 0.5$ m, like the VoD-P dataset. We choose $k_{\text{S}} = 8$ and $d_{\text{prox}} = 2$ m (based on Fig. 3(c)).



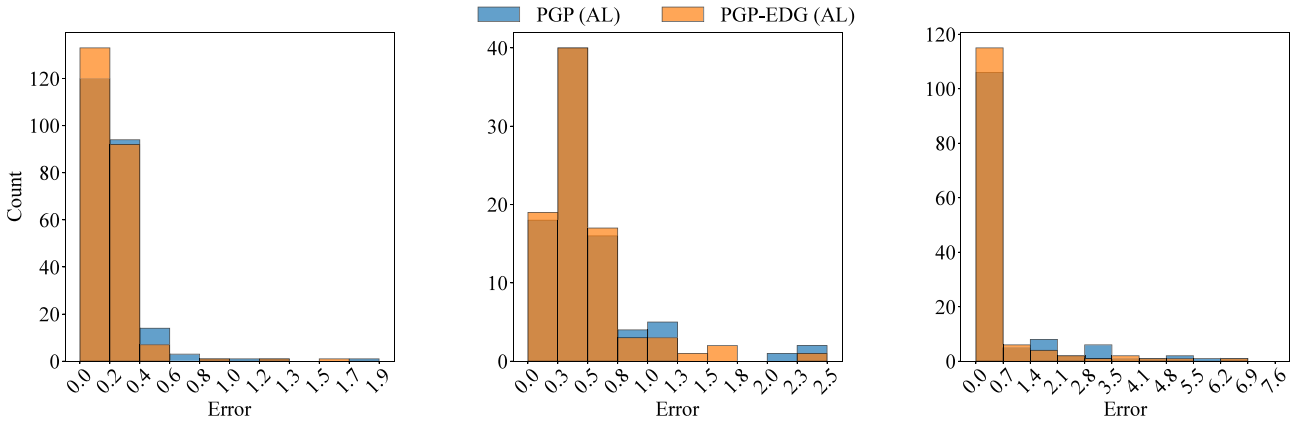
(a) Pedestrians (b) Cyclists (c) Vehicles

Fig. 5. Histogram of target agent velocities in the VoD-P annotated train, auto-labelled (AL) train, and (annotated) test set.

TABLE III
MEAN PERFORMANCE OVER THE VOD-P TEST SET WITH A $T_f = 3$ S PREDICTION HORIZON FOR $K = 10$ SAMPLES

Method	Pre-Trained on Auto-Labels (AL)	Pedestrian		Cyclist		Vehicle (Low-Speed)		Vehicle (High-Speed), OOD	
		minADE ↓	MR ↓	minADE ↓	MR ↓	minADE ↓	MR ↓	minADE ↓	MR ↓
KF [8]	☐	0.52	0.80	1.22	0.99	-	-	-	-
P2T [19]	☐	0.42	0.66	1.01	0.95	-	-	-	-
TNT* [10]	☐	0.33	0.54	0.78	0.95	0.43	0.63	5.60	1.00
TNT*-EDG	☐	0.29	0.49	0.74	0.92	0.39	0.60	4.30	1.00
PGP [2]	☐	0.27 ± 0.00	0.36 ± 0.01	0.60 ± 0.00	0.83 ± 0.03	0.33 ± 0.00	0.45 ± 0.01	$\mathbf{2.32 \pm 0.03}$	$\mathbf{0.92 \pm 0.04}$
PGP-D	☐	0.26 ± 0.00	0.34 ± 0.01	0.60 ± 0.01	0.80 ± 0.02	0.32 ± 0.00	0.44 ± 0.01	2.69 ± 0.04	0.94 ± 0.02
PGP-ED	☐	0.27 ± 0.00	0.37 ± 0.01	0.61 ± 0.01	0.81 ± 0.02	0.33 ± 0.00	0.46 ± 0.01	2.70 ± 0.02	0.98 ± 0.02
PGP-EDG	☐	$\mathbf{0.23 \pm 0.00}$	$\mathbf{0.29 \pm 0.01}$	$\mathbf{0.53 \pm 0.01}$	$\mathbf{0.76 \pm 0.04}$	$\mathbf{0.29 \pm 0.00}$	$\mathbf{0.40 \pm 0.01}$	2.46 ± 0.02	0.95 ± 0.03
TNT* [10]	☑	0.30	0.50	0.76	0.90	0.41	0.62	2.99	0.97
TNT*-EDG	☑	0.29	0.48	0.72	0.90	0.38	0.57	4.27	0.97
PGP [2]	☑	0.24 ± 0.00	0.31 ± 0.01	0.51 ± 0.01	$\mathbf{0.77 \pm 0.02}$	0.28 ± 0.00	0.38 ± 0.01	2.46 ± 0.03	0.97 ± 0.01
PGP-D	☑	0.23 ± 0.00	0.29 ± 0.01	0.59 ± 0.01	0.82 ± 0.01	0.30 ± 0.00	0.42 ± 0.01	1.80 ± 0.03	$\mathbf{0.92 \pm 0.02}$
PGP-ED	☑	0.22 ± 0.00	0.25 ± 0.02	0.58 ± 0.00	0.79 ± 0.01	0.29 ± 0.00	$\mathbf{0.37 \pm 0.01}$	$\mathbf{1.51 \pm 0.02}$	0.93 ± 0.04
PGP-EDG	☑	$\mathbf{0.21 \pm 0.00}$	$\mathbf{0.23 \pm 0.01}$	$\mathbf{0.48 \pm 0.01}$	$\mathbf{0.77 \pm 0.02}$	$\mathbf{0.27 \pm 0.00}$	$\mathbf{0.37 \pm 0.01}$	1.82 ± 0.03	0.97 ± 0.02

The mean and standard deviation over 10 evaluation runs are shown for the PGP [2] model and its variants. Best performance on each metric is shown in bold. minADE = minimum Average Displacement Error, MR = Miss Rate, OOD = Out-of-Distribution.



(a) Pedestrian (b) Cyclist (c) Vehicle

Fig. 6. Histogram of MinADE errors with $K = 10$ for the models pre-trained on auto-labels. Our best-performing model, PGP-EDG, has more predictions in the lower bins for each class. Challenging cases in the tail of the histogram remain.

B. *Quantitative Results*

See Table III for the quantitative experimental results. First, we consider the effect of pre-training the original PGP and TNT* models on auto-labelled data. A comparison between same models without and with pre-training (see above and below horizontal table separator) shows that the additional training data benefits the model's performance for all in-distribution classes.

We next consider the performance of the model variants described in Section III-C for each in-distribution agent class. We do not consider the Vehicle (High-Speed) subset here because it is difficult to draw conclusions on out-of-distribution data. We observe that our fully class-specific model PGP-EDG provides the best performance for each agent class, regardless of whether pre-training was used or not. This is further supported by Fig. 6, which shows the per-class histograms of minADE

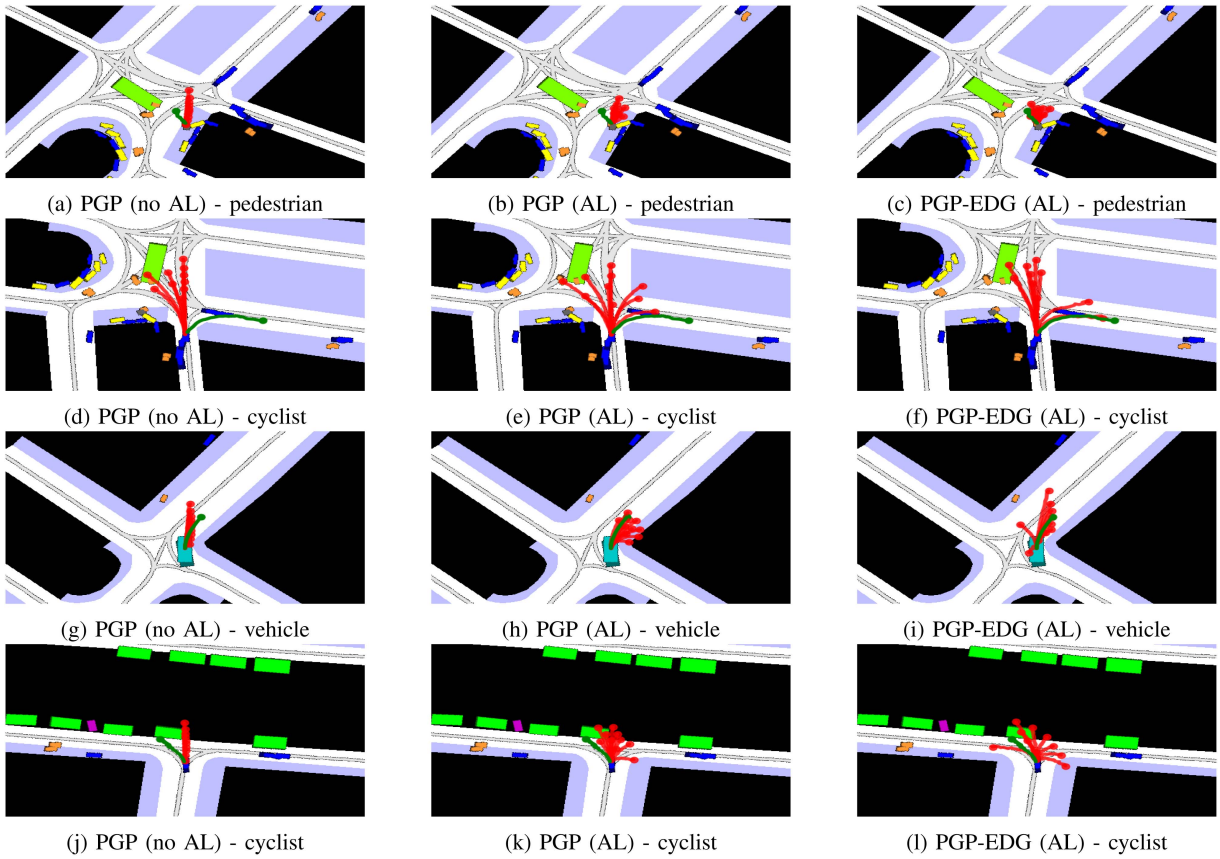


Fig. 7. Qualitative examples of model predictions on VoD-P dataset for $K = 10$ predictions. Predictions are shown in red; the ground truth future tracklet is shown in green. The baseline PGP [2] (no AL) model does not adequately account for multi-modality in its predictions, making wrong predictions in challenging cases e.g. at intersections; the models pre-trained on auto-labels (AL) perform better on these cases. PGP-EDG best accounts for the map-based intent and motion dynamics of each agent class. Failure cases - where no model accurately predicts the trajectory - remain (such as in the final row).

values for the baseline PGP model and PGP-EDG; there are more predictions in the lowest bin and fewer predictions in the tail of the histograms. Table III shows that TNT*-EDG also outperforms the baseline TNT* model across all classes. The greatest improvement occurs for pedestrians, with a 12.1% and 14.8% improvement in minADE for the TNT* and PGP models over the baselines (without pre-training), supporting that the new scene representation better covers pedestrian intent than the original PGP and TNT* representations. Vehicles benefit slightly more than cyclists from the class-specific model components, on average, showing that the dynamics of the classes are also better captured. Adding only some of the class-specific model components provides mixed results.

The best combination of modifications (i.e. PGP [2] without AL pre-training compared to PGP-EDG with AL pre-training) leads to a 22.2%, 20.0%, and 18.2% lower minADE, and 0.13, 0.06, and 0.08 point better MR for pedestrians, cyclists, and the low-speed vehicles, respectively, for $K = 10$ samples; the performance of TNT* on the minADE metric was improved by 12.1%, 11.1%, and 11.6%.

C. Qualitative Results

Fig. 7 shows examples of predictions from the baseline PGP [2] model, the pre-trained PGP model and the (pre-trained) PGP-EDG variant. In the top row, PGP (no AL) and PGP (AL) fail to predict that the pedestrian will take a sharp left onto the road; only PGP-EDG (AL) does so. Similarly, for the cyclist in the second row the PGP (AL) and PGP-EDG (AL) models predict that the cyclist may turn right, although the dynamics of the turn are predicted poorly by PGP (AL). Finally, both PGP (AL) and PGP-EDG (AL) accurately predict the right-hand turn of the vehicle in the bottom row that PGP (no AL) misses. However, a number of the predictions of PGP (AL) (incorrectly) end outside of the road. Remaining challenging cases include agents turning at intersections, and agents with high velocities. An example is shown in the final row; a cyclist pulls to the side of the road to avoid an oncoming vehicle (not shown) after turning into a one-way street. None of the models predict this behaviour.

D. Impact on Memory and Runtime

Table IV shows the model size and runtime for the TNT* [10] and PGP [2] models and their EDG variants, benchmarked on an Nvidia Titan V GPU. Although more resources are needed when using the class-specific EDG variants, the models are still feasible for real-time application. Also, transformer-based models such as the recent Wayformer [24] and MTR [1] models

TABLE IV
MODEL SIZE AND RUNTIME ON AN NVIDIA TITAN V GPU

	TNT* [10]	TNT*-EDG	PGP [2]	PGP-EDG	Wayformer [24]	MTR [1]
Parameters (M)	0.378	1.13	0.154	0.463	15.1	64.4
Memory (MB)	1.44	4.33	0.588	1.77	57.7	246
Runtime (ms/sample)	1.76	3.39	4.82	8.24	80.5	85.6

require many times more resources, highlighting the relative efficiency of even the EDG models.

V. CONCLUSION

We proposed class-specific map representations and model components to improve trajectory prediction in mixed traffic, especially in view of VRUs. We furthermore augmented the VoD-P dataset by auto-labelling the LiDAR pointclouds of the recording vehicle; these auto-labels are released as pre-training set. We investigated the effect of our proposed class-specific map representations and model components, as well as pre-training on the auto-labelled tracklets, and achieved a total improvement on the minADE metric of 22.2%, 20.0%, and 18.2% for the PGP model and 12.1%, 11.1%, 11.6% for the TNT model for pedestrians, cyclists, and (low-speed) vehicles, respectively. In future work, we plan to use additional criteria for grouping agents, such as speed profiles or interaction patterns.

REFERENCES

[1] S. Shi, L. Jiang, D. Dai, and B. Schiele, "Motion transformer with global intention localization and local movement refinement," in *Proc. Adv. Neural Inf. Process. Syst.*, 2022, vol. 35, pp. 6531–6543.

[2] N. Deo, E. Wolff, and O. Beijbom, "Multimodal trajectory prediction conditioned on lane-graph traversals," in *Proc. Conf. Robot. Learn.*, 2022, pp. 203–212.

[3] Z. Sun, Z. Wang, L. Halilaj, and J. Luettin, "SemanticFormer: Holistic and semantic traffic scene representation for trajectory prediction using knowledge graphs," *IEEE Robot. Automat. Lett.*, vol. 9, no. 9, pp. 7381–7388, Sep. 2024.

[4] Z. Lan, Y. Jiang, Y. Mu, C. Chen, and S. E. Li, "SEPT: Towards efficient scene representation learning for motion prediction," in *Proc. 12th Int. Conf. Learn. Representations*, 2023.

[5] H. Caesar et al., "nuScenes: A multimodal dataset for autonomous driving," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 11618–11628.

[6] S. Ettinger et al., "Large scale interactive motion forecasting for autonomous driving: The Waymo open motion dataset," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2021, pp. 9690–9699.

[7] B. Wilson et al., "Argoverse 2: Next generation datasets for self-driving perception and forecasting," in *Proc. 35th Conf. Neural Inf. Process. Syst. Datasets Benchmarks Track*, 2024.

[8] H. J. Boekema, B. K. Martens, J. F. Kooij, and D. M. Gavrila, "Multi-class trajectory prediction in urban traffic using the View-of-Delft Prediction dataset," *IEEE Robot. Automat. Lett.*, vol. 9, no. 5, pp. 4806–4813, May 2024.

[9] A. Palffy, E. Pool, S. Baratam, J. F. Kooij, and D. M. Gavrila, "Multi-class road user detection with 3+1D radar in the View-of-Delft dataset," *IEEE Robot. Automat. Lett.*, vol. 7, no. 2, pp. 4961–4968, Apr. 2022.

[10] H. Zhao et al., "TNT: Target-driven trajectory prediction," in *Proc. Conf. Robot. Learn.*, 2021, pp. 895–904.

[11] J. Liu, X. Mao, Y. Fang, D. Zhu, and M. Q.-H. Meng, "A survey on deep-learning approaches for vehicle trajectory prediction in autonomous driving," in *Proc. 2021 IEEE Int. Conf. Robot. Biomimetics*, 2021, pp. 978–985.

[12] A. Rudenko, L. Palmieri, M. Herman, K. M. Kitani, D. M. Gavrila, and K. O. Arras, "Human motion trajectory prediction: A survey," *Int. J. Robot. Res.*, vol. 39, no. 8, pp. 895–935, 2020.

[13] P. Karle, M. Geisslinger, J. Betz, and M. Lienkamp, "Scenario understanding and motion prediction for autonomous vehicles—Review and comparison," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 10, pp. 16962–16982, Oct. 2022.

[14] 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.

[15] M. Golchoubian, M. Ghafurian, K. Dautenhahn, and N. L. Azad, "Pedestrian trajectory prediction in pedestrian-vehicle mixed environments: A systematic review," *IEEE Trans. Intell. Transp. Syst.*, vol. 24, no. 11, pp. 11544–11567, Nov. 2023.

[16] J. Gu et al., "DenseTNT: End-to-end trajectory prediction from dense goal sets," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2021, pp. 15283–15292.

[17] T. Gilles, S. Sabatini, D. Tsishkou, B. Stanciulescu, and F. Moutarde, " GOHOME: Graph-oriented heatmap output for future motion estimation," in *Proc. 2022 Int. Conf. Robot. Automat.*, 2022, pp. 9107–9114.

[18] W. Zeng, M. Liang, R. Liao, and R. Urtasun, "LaneRCNN: Distributed representations for graph-centric motion forecasting," in *Proc. 2021 IEEE/RSJ Int. Conf. Intell. Robot. Syst.*, 2021, pp. 532–539.

[19] N. Deo and M. M. Trivedi, "Trajectory forecasts in unknown environments conditioned on grid-based plans," 2020, *arXiv:2001.00735*.

[20] C. Feng et al., "MacFormer: Map-agent coupled transformer for real-time and robust trajectory prediction," *IEEE Robot. Autom. Lett.*, vol. 8, no. 10, pp. 6795–6802, Oct. 2023.

[21] J. Gao et al., "VectorNet: Encoding HD maps and agent dynamics from vectorized representation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 11522–11530.

[22] M. Liang et al., "Learning lane graph representations for motion forecasting," in *Proc. 16th Eur. Conf. Comput. Vis.*, Glasgow, U.K.: Springer, 2020, pp. 541–556.

[23] D. Park, H. Ryu, Y. Yang, J. Cho, J. Kim, and K.-J. Yoon, "Leveraging future relationship reasoning for vehicle trajectory prediction," in *Proc. 11th Int. Conf. Learn. Representations*, 2023.

[24] N. Nayakanti, R. Al-Rfou, A. Zhou, K. Goel, K. S. Refaat, and B. Sapp, "Wayformer: Motion forecasting via simple & efficient attention networks," in *Proc. 2023 IEEE Int. Conf. Robot. Automat.*, 2023, pp. 2980–2987.

[25] Y. Guan et al., "Integrated decision and control: Toward interpretable and computationally efficient driving intelligence," *IEEE Trans. Cybern.*, vol. 53, no. 2, pp. 859–873, Feb. 2023.

[26] A. H. Lang, S. Vora, H. Caesar, L. Zhou, J. Yang, and O. Beijbom, "PointPillars: Fast encoders for object detection from point clouds," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 12689–12697.

[27] Q. Wang, Y. Chen, Z. Pang, N. Wang, and Z. Zhang, "Immortal tracker: Tracklet never dies," 2021, *arXiv:2111.13672*.