InsMapper: Exploring Inner-instance Information for Vectorized HD Mapping

Zhenhua Xu Kwan-Yee K. Wong Hengshuang Zhao*

The University of Hong Kong

Project webpage: https://tonyxuqaq.github.io/InsMapper

Abstract. Vectorized high-definition (HD) maps contain detailed information about surrounding road elements, which are crucial for various downstream tasks in modern autonomous vehicles, such as motion planning and vehicle control. Recent works attempt to directly detect the vectorized HD map as a point set prediction task, achieving notable detection performance improvements. However, these methods usually overlook and fail to analyze the important inner-instance correlations between predicted points, impeding further advancements. To address this issue, we investigate the utilization of inner-instance information for vectorized high-definition mapping through transformers, and propose a powerful system named **InsMapper**, which effectively harnesses inner-instance information with three exquisite designs, including hybrid query generation, inner-instance query fusion, and inner-instance feature aggregation. The first two modules can better initialize queries for line detection, while the last one refines predicted line instances. InsMapper is highly adaptable and can be seamlessly modified to align with the most recent HD map detection frameworks. Extensive experimental evaluations are conducted on the challenging NuScenes and Argoverse 2 datasets, where InsMapper surpasses the previous state-of-the-art method, demonstrating its effectiveness and generality.

Keywords: Autonomous Driving · High-definition Map

1 Introduction

Vectorized high-definition maps (HD maps) play a critical role in today's autonomous vehicles [6, 23, 27], as they contain detailed information about the road, including the position of road elements (*e.g.*, road boundaries, lane splits, pedestrian crossings, and lane centerlines), connectivity, and topology of the road. Without the assistance of HD maps for perceiving and understanding road elements, unexpected vehicle behaviors may be encountered, such as incorrect path planning results or even vehicle collisions.

Typically, HD maps are created by offline human annotation, which is labor-intensive, inefficient, and expensive. Although there are works proposing to make

* Corresponding author.

such an offline process automatic [38, 40, 41], it is not possible to recreate and update the HD map frequently when the road has been modified, such as when a new road is built or an existing road is removed. To address this issue, several recent studies propose to detect local HD maps using an online method [7, 20, 24, 25, 27, 32]. This paper aims to further investigate online HD map detection techniques based on vehicle-mounted sensors (*i.e.*, cameras and LiDARs).

Early online HD map detection works consider road element mapping as a semantic segmentation task in bird's-eye view (BEV) [22, 30, 31], in which road elements are predicted in raster format (*i.e.*, pixel-level semantic segmentation mask). However, rasterized road maps cannot be effectively utilized by downstream tasks of autonomous vehicles, such as motion planning and vehicle control [23, 26, 27]. Moreover, it is challenging to distinguish instances from the rasterized map, especially when some road elements overlap with each other or when a complicated topology is encountered (*e.g.*, road split or road intersections). To alleviate these problems, HDMapNet [20] proposes hand-crafted post-processing algorithms to better obtain road element instances for vectorized HD maps. However, HDMapNet still heavily relies on rasterized results, restricting it from handling complicated urban scenes. Recently, some works have resorted to predicting vectorized HD maps directly [3, 24, 27, 35]. A two-stage hierarchical set prediction method is proposed in VectorMapNet [27]. After predicting key points of road elements in the HD map, VectorMapNet sequentially generates intermediate points. Although VectorMapNet is considered to be the first online vectorized HD map detection work, the sequential operation degrades its efficiency and model performance. MapTR [24] further proposes to use DETR [5, 46] for vectorized HD map detection as a point set prediction problem, and the output point sets are then grouped into road element instances. Some recent works further improve the performance of MapTR [7, 25, 32]. Among them, MapTR-V2 achieves state-of-the-art performance, but it overlooks and fails to utilize the essential inner-instance correlations between points to further boost the performance.

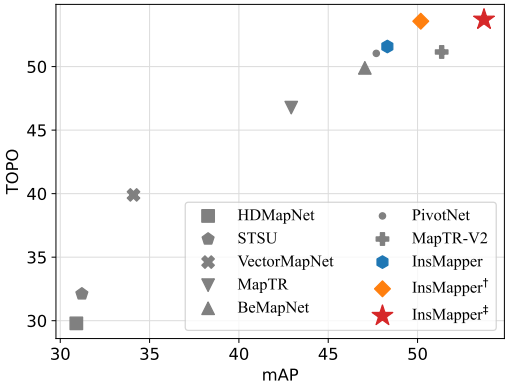


Fig. 1: Comparison of vectorized HD map detection methods. All solutions are evaluated on the NuScenes validation set. The x-axis represents the mAP results and the y-axis displays topology level correctness [13]. InsMapper is flexible and adaptable, so it can be seamlessly modified to align with multiple existing frameworks. InsMapper is built on MapTR [24]. † means InsMapper based on more recent framework PivotNet [7], while ‡ indicates InsMapper based on previous SOTA, MapTR-V2 [25].

In this work, we focus on investigating the utilization of inner-instance information for vectorized high-definition mapping via transformer, and propose a new framework named **InsMapper**, which can utilize inner-instance point information for improved online HD map detection via three exquisite designs. First, a hybrid query generation module is introduced to better initialize queries for detection. Then, an inner-instance query fusion module is added before the transformer decoder, which fuses inner-instance object queries to refine the initialized line instances. Finally, a transformer-based inner-instance aggregation module is incorporated to further refine the predicted lines. Extensive experimental evaluations are conducted on the popular and challenging NuScenes [2] and Argoverse 2 [37] datasets. InsMapper is based on MapTR [24], and it can be seamlessly modified based on the latest frameworks with more enhanced performance, such as PivotNet [7] and MapTR-V2 [25]. No matter what the base framework is, InsMapper harvests state-of-the-art performance, surpassing the previous highest solutions. The intuitive performance comparisons are illustrated in Figure 1, and our main contributions can be summarized as follows:

- We analyze the limitations of existing HD map detection algorithms and propose a new framework named InsMapper, which can effectively utilize the easily overlooked inner-instance point correlation information for accurate and generalizable online HD map detection.
- We incorporate three useful modules to leverage inner-instance information, including hybrid query generation, inner-instance query fusion, and inner-instance feature aggregation. The first two modules better initialize queries for detection and the last one refines detected line instances.
- We conduct experiments on the challenging NuScenes and Argoverse 2 datasets. InsMapper outperforms all baselines by a large margin. The achieved state-of-the-art results demonstrate its effectiveness and generality.

2 Related Work

Vector map detection. Vector maps use vectors to represent road elements, such as road networks, road curbs, and lane lines. A high-definition map (HD map) is a type of vector map, but a vector map may not qualify as an HD map if it lacks sufficient resolution or fails to provide lane-level information, among other factors. Online HD map detection can benefit from insights gained from a broader scope of vector map detection works, including road-network detection [1,14,39, 41], road-curb detection in aerial images [42, 43], and road lane line detection [15, 16], etc. DagMapper [16] aims to detect vectorized lane lines from pre-built point cloud maps using iterations. However, DagMapper only handles simple topology changes of lane lines and struggles to handle complex road intersections. RNGDet++ [41] applies DETR along with an imitation learning algorithm [34] for road-network detection, achieving state-of-the-art performance on multiple datasets. Although these methods achieve satisfactory graph detection accuracy, they suffer from poor efficiency due to their iterative algorithms. Given the

stringent real-time requirements of autonomous vehicles, they are not suitable for online HD map detection.

HD map detection. HD map detection was initially a subtask of BEV detection [21, 22, 28, 30, 31]. Recently, given the importance of HD maps, several works have directly focused on HD map detection [4, 20, 24, 27, 29, 33, 35, 38, 40]. However, most of these works either involve offline HD map creation [12, 38, 40] or only detect one specific road element [3, 4, 33]. HDMapNet [20] is considered to be the first work specifically designed for multiple road element detection. However, HDMapNet outputs rasterized results, requiring complicated handcrafted post-processing to obtain vectorized HD maps. To address this issue, VectorMapNet [27] is believed to be the first work detecting vectorized HD maps in real-time. However, it consists of two stages, and its efficiency is significantly impacted by sequential operations. In contrast, MapTR [24] uses deformable DETR [46] to design an end-to-end vectorized HD map detection model, which greatly simplifies the pipeline and delivers better detection performance. Some recent works further improve the performance of MapTR [7,25,32]. Among them, MapTR-V2 achieves state-of-the-art performance on vectorized HD map detection, but it still overlooks and does not investigate the inner-instance correlations between points, limiting further improvements.

Detection by Transformer. DETR [5] is the first end-to-end transformer-based object detection framework. Compared with previous CNN-based methods [8, 10], DETR removes the need for anchor proposals and non-maximum suppression (NMS), making it simpler and more effective. To address the issue of slow convergence, subsequent works propose accelerating DETR training through deformable attention [46], denoising [18], and dynamic query generation [19, 36, 45]. Intuitively, most of these DETR refinements could be adapted to the HD map detection task since our proposed InsMapper relies on DETR. However, unlike typical object detection tasks in which objects are assumed to be independent and identically distributed (i.i.d.), detected points in HD maps have strong correlations, especially among inner-instance points. This inherent difference makes some refined DETR methods unsuitable to our task and additional investigations are required.

3 Point Correlation

3.1 Vector Map Decomposition and Sampling

Let G denote the original vector map label of the scene, which consists of vertices V and edges E. The vector map contains multiple classes of road elements, including pedestrian crossings, road dividers, road boundaries, and lane centerlines. Among them, the first three classes of road elements are simple polylines or polygons without intersection points. While lane centerlines have more complicated topology, such as lane split, lane merge, and lane intersection. To unify all vector elements, the vector map is decomposed into simpler shapes (*i.e.*, polylines and polygons) without intersections. Any vertices in the vector map with degrees larger than two (*i.e.*, intersection vertices) are removed from G, but

incident edges are kept. In this way, a set of simple polylines and polygons without intersections is obtained as $G^* = \{l_i\}_{i=1}^{N^*}$, where G^* is an undirected graph. Each shape l_i is defined as an instance, and N^* denotes the overall number of instances in a vector map.

Following MapTR and MapTR-V2, to enhance the parallelization capacity of the model, each instance is evenly re-sampled with fixed-length points as $l_i = (v_1, ..., v_j, ..., v_{n_p})$. l_i is ordered from v_1 to v_{n_p}, with n_p being the number of sampled points for each instance. For polygon instances, v_1 is equal to v_{n_p}. Please refer to the supplementary document for the visualization of the pre-processing.

3.2 Inner-instance Correlation

Unlike conventional object detection tasks, where objects can be approximated as independent and identically distributed (i.i.d.), strong correlations exist between points within the same line instance in the vectorized HD map detection task. Please refer to the supplementary document for the visualization of the point correlation.

Inner-instance correlation is crucial for point coordinates prediction. Points within the same instance can collaborate by sharing inner-instance information, leading to smoother and more accurate predictions. Without this collaboration, points of the same instance may produce independent predictions, leading to zigzag or incorrect instances. The inner-instance correlation can serve as additional constraints to facilitate the query initialization and refinement of predicted lines. In previous methods, the correlation of points is not correctly analyzed and leveraged, limiting further improvement. In subsequent sections, we introduce InsMapper to more effectively utilize point correlations and improve vectorized map detection performance.

4 Methodology

4.1 Overall Framework

Building on MapTR [24], our proposed InsMapper is an end-to-end trainable framework for online vectorized HD map detection, as illustrated in Figure 2.

InsMapper is adaptive and can be seamlessly modified based on the latest frameworks with more enhanced performance, such as PivotNet [7] and MapTR-V2 [25]. Following BEVformer [22], InsMapper first projects perspective view camera images into the bird's-eye-view (BEV). After obtaining the BEV features, InsMapper uses deformable attention layers [46] to process input object queries. Each object query predicts one point, including the object class and the regression point position. To better leverage inner-instance information and further enhance the final detection performance, we introduce three designs in InsMapper: hybrid query generation, which replaces the hierarchical query in MapTR, generating line queries with better diversity; inner-instance query fusion, which fuses generated queries within the same instance based on

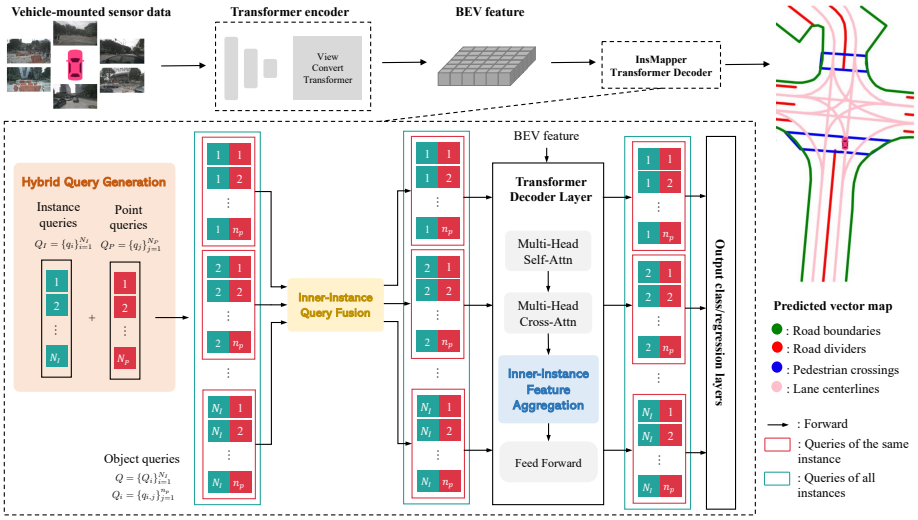


Fig. 2: Overall framework. InsMapper is an end-to-end transformer model with an encoder-decoder structure. The transformer encoder projects perspective-view camera images into a bird's-eye view (BEV). Subsequently, the transformer decoder detects vector instances by predicting point sets. To enhance the utilization of inner-instance information, we introduce the following three components: a hybrid query generation scheme (orange module), an inner-instance query fusion module (yellow module), and an inner-instance feature aggregation module (blue module). The first two modules better initialize queries for detection and the final one refines detected line instances.

inner-instance features to further enhance the quality of initialized line queries; and inner-instance feature aggregation, which adds a new inner-instance self-attention module to decoder layers to further aggregate inner-instance features for line refinement.

Different from past works, which treat queries as independently distributed, InsMapper utilizes inner-instance information to fuse object queries and aggregate intermediate features in deformable transformer decoder layers. Otherwise, adjacent vertices may produce independent position regression results, leading to zigzag or even incorrect instances. With the assistance of inner-instance information, the model can better initialize queries for detection and refine the positions of detected points within an instance, significantly enhancing the final performance, as demonstrated in the experiments section.

4.2 Query Generation

In contrast to traditional object detection tasks, where queries are independent and identically distributed (i.i.d.), the points to be detected in our task exhibit significant correlations. Various methods can be used to generate object queries as input for the transformer decoder, such as the basic scheme, hierar-

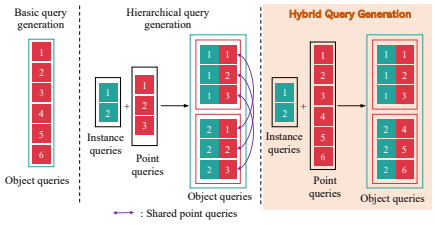
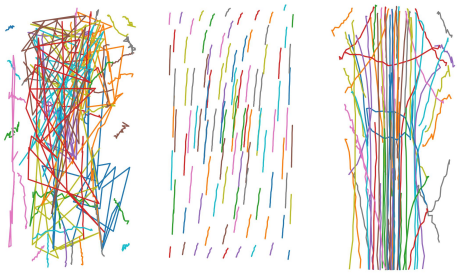


Fig. 3: Query generation schemes. For concise visualization, the number of instances N_I is 2, and the number of points per instance n_p is 3. The hybrid scheme can initialize lines with both good diversity and quality.



(a) Basic **(b)** Hierarchical **(c)** Hybrid

Fig. 4: Visualization of initialized lines with different query generation schemes. The quality and diversity of initialized lines are crucial for the final performance. Each color represents one initialized line.

chical scheme, and hybrid scheme. All query generation schemes are illustrated in Figure 3. Initialized queries are organized into line instances. The quality and diversity of the initialized lines are both crucial.

Basic query generation. This method is leveraged in the original DETR [5] for query generation. It assumes that points are i.i.d. and generates queries randomly, without utilizing inner-instance information. Let N_I denote the number of predicted instances, which is obviously larger than N^*. Since each instance contains n_p points, the basic object queries consist of $N_I \cdot n_p$ randomly generated i.i.d. queries, resulting in irregular line shapes. Due to the lack of constraints on inner-instance points, the initialized line instances are noisy and have poor quality, which leads to degraded performance. MapTR with the basic query scheme is visualized in Figure 4a.

Hierarchical query generation. To address the aforementioned issue of the basic scheme, a hierarchical query generation scheme is proposed in MapTR [24] and MapTR-V2 [25]. Let $Q_{ins} = \{q_i^I\}_{i=1}^{N_I}$ denote instance queries and $Q_{pts} = \{q_j^P\}_{j=1}^{n_p}$ denote point queries. The object queries for HD map detection are then the pairwise addition of Q_{ins} and Q_{pts}:

$$Q = \{q_{i,j} = q_i^I + q_j^P | q_i^I \in Q_{ins}, q_j^P \in Q_{pts}\}, \tag{1}$$

where $|Q| = |Q_{ins}| \cdot |Q_{pts}| = N_I \cdot n_p$. For each line instance i, the instance query q_i^I denotes its uniqueness. By using q_i^I as a bridge, object queries within the same instance can better collaborate with each other for point prediction. But all line instances share the same point queries so all initialized lines tend to have quite similar shapes, degrading initialization diversity, which harms the final detection performance. The initialized lines generated by MapTR with the hierarchical query scheme are visualized in Figure 4b.

Hybrid query generation. The basic scheme does not take into account any information exchange between queries, whereas the hierarchical one generates

repetitive line shapes. To address these issues, this paper presents a hybrid query generation method that mitigates the drawbacks of the aforementioned schemes while maintaining appropriate inner-instance information exchange.

Let the instance queries be $Q_{ins} = \{q_i^I\}_{i=1}^{N_I}$, and the point queries be $Q_{pts} = \{q_j^P\}_{j=1}^{N_P} = \{q_j^P\}_{j=1}^{N_I \cdot n_p}$. The point queries are divided into N_I instance groups, and a point query q_j^P is assigned to the $\lceil \frac{j}{n_p} \rceil$-th instance. Consequently, the final object query is the sum of a point query p_j^P and its assigned instance query p_k^I, where $k = \lceil \frac{j}{n_p} \rceil$. The object query set can be expressed as:

$$Q = \{q_k^I + q_j^P | q_j^P \in Q_{pts}\} \tag{2}$$

where $|Q| = |Q_{pts}| = N_P = N_I \cdot n_p$. In contrast to the hierarchical scheme, each point query in the hybrid scheme is utilized only once to generate the object query, thereby preventing repetitive line shapes. Simultaneously, point queries belonging to the same instance are summed with a shared instance query, establishing the inner-instance connection. The hybrid query generation method can be considered as a combination of the basic and hierarchical schemes, effectively mitigating their respective drawbacks. InsMapper with the hybrid query generation scheme is visualized in Figure 4c.

4.3 Inner-instance Query Fusion

The input object queries for the transformer decoder are generated from instance queries and point queries. Although the generated queries can leverage some inner-instance information, this information exchange is indirect and inflexible. The instance query is distributed equally among all inner-instance points, while a more precise point-to-point information exchange cannot be realized. As a result, a query fusion module is introduced to further utilize inner-instance information and refine initialized lines.

Let $Q_i = \{q_{i,j}\}_{j=1}^{n_p}$ represent the set of object queries belonging to the i-th instance, and $q_{i,j}$ denote the j-th point of the i-th instance. $q_{i,j}$ is correlated with all other queries in Q_i. To better fuse inner-instance object queries, each query is updated by a weighted sum of all queries in Q_i as:

$$q_{i,j} = f(q_{i,j}, Q_i) = \sum_{k=1}^{n_p} w_{i,j,k} \phi(q_{i,k}) \tag{3}$$

where $w_{i,j,k}$ demonstrates weights for query fusion. $f(\cdot)$ is the fusion function and $\phi(\cdot)$ is the kernel in case nonlinear transformation is needed. $f(\cdot)$ could be realized by handcraft weights, fully connected layers, or self-attention. In this work, self-attention is applied as $f(\cdot)$.

In conventional object detection tasks, object queries are assumed to be independent and identically distributed (i.i.d.), making query fusion unnecessary. However, in the task of HD map detection, the query fusion module effectively aligns the update of queries and enables each point to pay more attention to

(a) No fusion **(b)** With fusion

Fig. 5: Visualization of the effect of the inner-instance query fusion module. (a) No query fusion. (b) With query fusion. The query fusion module can better refine the initialized lines based on inner-instance information. In this example, the proposed query fusion module significantly enhances the diversity of initialized line instances.

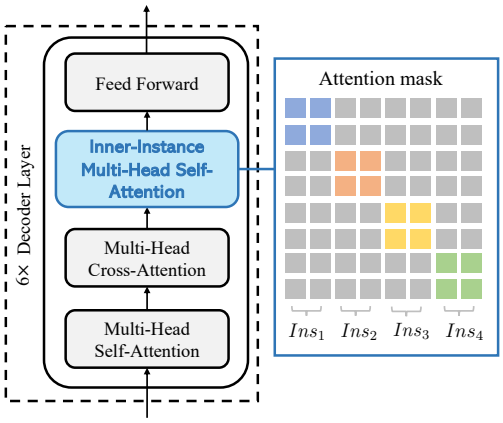


Fig. 6: Decoder of InsMapper. An inner-instance multi-head self-attention module is incorporated into decoder layers. In this module, the attention between points belonging to different instances is blocked (grey grids). Only inner-instance attention is allowed (colored grids). Colored grids are randomly blocked with ϵ probability (set to grey) for robustness. The proposed module can further utilize inner-instance information and refine the predicted lines.

neighboring points. It can greatly refine the initialized lines for better query initialization, leading to enhanced performance. Without this module, queries within the same instance cannot be aware of each other, rendering them "blind". Poor line initialization tends to have degraded detection scores. The effect of the query fusion module is visualized in Figure 5.

4.4 Inner-instance Feature Aggregation

In addition to object query manipulation, InsMapper performs inner-instance feature aggregation for predicted line refinement in the transformer decoder layers by incorporating an additional inner-instance self-attention module. Inspired by [9], inner-instance attentions are randomly blocked with probability ϵ for robustness. The decoder layer of InsMapper is depicted in Figure 6.

The inner-instance self-attention module resembles the original self-attention module but features a specially designed attention mask. As illustrated in Figure 6, the attention mask of inner-instance points is set to zero (colored grids), indicating that attention between points within the same instance is allowed. Conversely, for points belonging to different instances (*i.e.*, inter-instance points), the corresponding attention mask values are set to one, blocking attention between them. This method encourages the model to focus more on inner-instance

information, resulting in more consistent predictions. To further enhance the robustness of this module, the inner-instance attention (colored grids) has an ϵ probability of being blocked.

An alternative method involves placing the proposed inner-instance attention module before the cross-attention module. However, the self-attention module before cross-attention should not block inter-instance information. Otherwise, some instances may produce duplicate predictions since they cannot "clearly see" other instances, which is analyzed in the original DETR [5]. Consequently, implementing inner-instance self attention before the cross-attention module leads to degraded final performance. Thus the proposed inner-instance feature aggregation should be placed after the cross-attention layer. More experiments about the decoder structure are provided in the appendix.

5 Experiments

5.1 Experimental Setup

Datasets. Extensive experiments are conducted on the popular and challenging NuScenes dataset [2] and the Argoverse 2 dataset [37]. These two datasets comprise hundreds of data sequences captured in various autonomous driving scenarios, encompassing diverse weather conditions and time periods. The perception range for the X-axis and Y-axis in the BEV is set to $[-15m, 15m]$ and $[-30m, 30m]$, respectively. The detected vectorized HD map should encompass four types of road elements simultaneously: road boundary, lane split, pedestrian crossing, and lane centerline. Among these, the lane centerline is a crucial element for ensuring effective vehicle planning and control. Detecting lane centerlines poses greater difficulty compared to the other three elements.

Implementation details. We perform all the experiments on a machine equipped with 8 RTX-3090 GPUs. During the training phase, all GPUs are utilized, whereas only a single GPU is employed for inference. For our proposed InsMapper, we adopt multiple settings akin to the previous SOTA method, MapTR. All ablation studies are conducted for 24 epochs, with ResNet50 [11] as the backbone. For a fair comparison, except for the proposed modules, we keep the experiment settings exactly the same, such as batch size, network depth, the number of input queries, etc.

Evaluation metrics. Models are assessed using two types of metrics. The first metric, average precision (AP), gauges the instance-level detection performance, employing Chamfer distance for matching predictions with ground truth labels. To ensure a fair comparison, we follow previous works to calculate multiple AP_τ values with $\tau \in \{0.5, 1.0, 1.5\}$, and report the average AP. To measure the topology correctness, the metric TOPO that is widely used in past works [12–14, 41] is reported. For all metrics, a larger value indicates better performance.

5.2 Performance Comparison

In this section, we compare InsMapper with previous state-of-the-art methods using the aforementioned evaluation metrics. The quantitative comparison results

Table 1: Quantitative results of comparison experiments on the NuScenes validation set. Colored numbers show differences between InsMapper and the SOTA baseline under the same experiment setting. "C" and "L" denote camera and LiDAR, respectively. "-" denotes that the result is not available. "V2-99" and "Sec" correspond to VoVNetV2-99 [17] and SECOND [44]. † represents InsMapper based on PivotNet [7], ‡ indicates InsMapper based on MapTR-V2 [25].

Methods	Epochs	Backbone	Modality	AP_{ped}	AP_{div}	AP_{bound}	AP_{center}	mAP	TOPO
HDMapNet	30	Effi-B0	C	4.41	23.73	58.17	37.29	30.90	29.79
STSU	110	R50	C	-	-	-	31.21	31.21	32.11
VectorMapNet	130	R50	C	28.66	39.74	33.06	34.93	34.10	39.90
MapTR	24	R50	C	37.92	46.25	50.07	37.47	42.93	46.77
MapTR	24	V2-99	C	44.57	56.25	60.58	46.30	51.92	55.16
MapTR	24	R50&Sec	C&L	49.47	58.98	66.72	44.89	55.01	56.77
MapTR	110	R50	C	49.16	59.12	58.93	47.26	53.62	59.16
BeMapNet	24	R50	C	43.59	54.11	51.90	38.60	47.05	49.90
PivotNet	24	R50	C	42.73	53.55	52.01	42.47	47.69	51.04
MapTR-V2	24	R50	C	48.20	54.75	56.66	45.86	51.37	51.15
MapTR-V2	110	R50	C	58.68	65.72	67.12	56.16	61.92	63.26
InsMapper	24	R50	C	44.36	53.36	52.77	42.35	48.31	51.58
InsMapper	24	V2-99	C	51.16	63.71	64.47	51.40	57.68	60.00
InsMapper	24	R50&Sec	C&L	56.00	63.42	71.61	52.85	60.97	62.51
InsMapper	110	R50	C	55.42	63.87	65.80	54.20	59.40	66.19
InsMapper†	24	R50	C	46.45	54.91	54.16	45.20	50.18	53.57
InsMapper‡	24	R50	C	49.07	57.98	59.84	47.99	53.72	53.00
InsMapper‡	110	R50	C	**62.09**	**67.60**	**68.15**	**58.41**	**64.06**	**64.72**

Table 2: Quantitative results of comparison experiments on the Argoverse 2 validation set. ‡ indicates InsMapper based on MapTR-V2 [25].

Methods	Epochs	Backbone	Modality	AP_{ped}	AP_{div}	AP_{bound}	AP_{center}	mAP	TOPO
MapTR	6	R50	C	52.88	63.68	61.18	59.73	59.37	75.79
MapTR-V2	6	R50	C	57.16	67.96	65.25	63.20	63.39	77.94
InsMapper	6	R50	C	55.61	66.60	62.58	62.67	61.87	77.58
InsMapper‡	6	R50	C	**58.91**	**71.35**	**66.90**	**64.70**	**65.46**	**79.26**

Table 3: Quantitative results of comparison experiments on the NuScenes validation set without centerline. ‡ indicates InsMapper based on MapTR-V2 [25].

Methods	Epochs	Backbone	Modality	AP_{ped}	AP_{div}	AP_{bound}	mAP
MapTR	24	R50	C	46.04	51.58	53.08	50.23
MapTR-V2	24	R50	C	59.80	62.40	62.40	61.50
InsMapper	24	R50	C	48.44	54.68	56.92	53.35
InsMapper‡	24	R50	C	**61.54**	**65.09**	**64.62**	**63.75**

are reported in Table 1. InsMapper is flexible and adaptive, making it seamless to modify it based on existing transformer frameworks. InsMapper is built on MapTR for experiments, we also report the results of InsMapper with different base frameworks, *e.g.* InsMapper† is based on PivotNet [7] and InsMapper‡ relies on the previous SOTA method MapTR-V2 [25]. Compared to the previous methods, InsMapper improves the AP of all road elements by around 2. The topological metric also sees an improvement of around 1.5. The inference FPS of

Table 4: Quantitative results of comparison experiments on the NuScenes validation set. Centerlines are directed graphs in this table. ‡ indicates InsMapper based on MapTR-V2 [25].

Methods	Epochs	Backbone	Modality	AP_{ped}	AP_{div}	AP_{bound}	AP_{center}	mAP	TOPO
MapTR	24	R50	C	37.39	46.61	50.69	37.22	42.98	46.98
MapTR-V2	24	R50	C	49.53	55.46	58.48	53.82	54.32	54.90
InsMapper	24	R50	C	43.24	53.77	53.89	43.22	48.53	52.09
InsMapper‡	24	R50	C	**51.62**	**57.88**	**60.90**	**56.52**	**56.73**	**56.01**

MapTR and InsMapper are 8.5 and 7.5, respectively. The evaluation outcomes on the Argoverse 2 dataset [37] can be found in Table 2. InsMapper outperforms past works with notable enhancements. Table 3 presents the results without centerlines. InsMapper with different bases attains an improvement of more than 2 mAP. In the aforementioned experiments, all road elements are treated as undirected graphs. Table 4 shows the results when centerlines are represented as directed graphs. Based on the results, InsMapper still presents superior results than past works for directed HD map detection. Furthermore, we illustrate the qualitative visualizations in Figure 7. Owing to the effective employment of inner-instance information, InsMapper generates smoother and more precise HD maps compared to previous works.

Consequently, InsMapper demonstrates superior performance compared to prior approaches across multiple datasets, exhibiting outstanding detection outcomes. Concurrently, InsMapper maintains a comparable inference speed to the previous state-of-the-art method. These exceptional experimental results strongly demonstrate the effectiveness and generality of the proposed designs.

5.3 Ablation Studies

Key component designs. We conduct several ablation study experiments to confirm the necessity of the proposed modules. Initially, we incorporate these modules into the previous state-of-the-art baseline model incrementally, and the results are presented in Table 5. It is evident that all three modules contribute to performance improvements, thereby validating their necessity. To further investigate the impact of each module's design, we perform comprehensive ablation studies in the following paragraphs.

Hybrid query generation. Compared to basic or hierarchical query generation schemes, the proposed hybrid query generation scheme generates queries with better quality and diversity, due to effectively utilizing inner-instance correlation. The evaluation results of the different query generation schemes are presented in Table 6. These results reveal that the hybrid query generation scheme outperforms its counterparts, thereby demonstrating the soundness of this design.

Inner-instance query fusion. Query fusion is proposed to leverage the correlation between inner-instance queries to enhance prediction performance. Various methods can be employed for fusing queries, including mean fusing (*i.e.*,

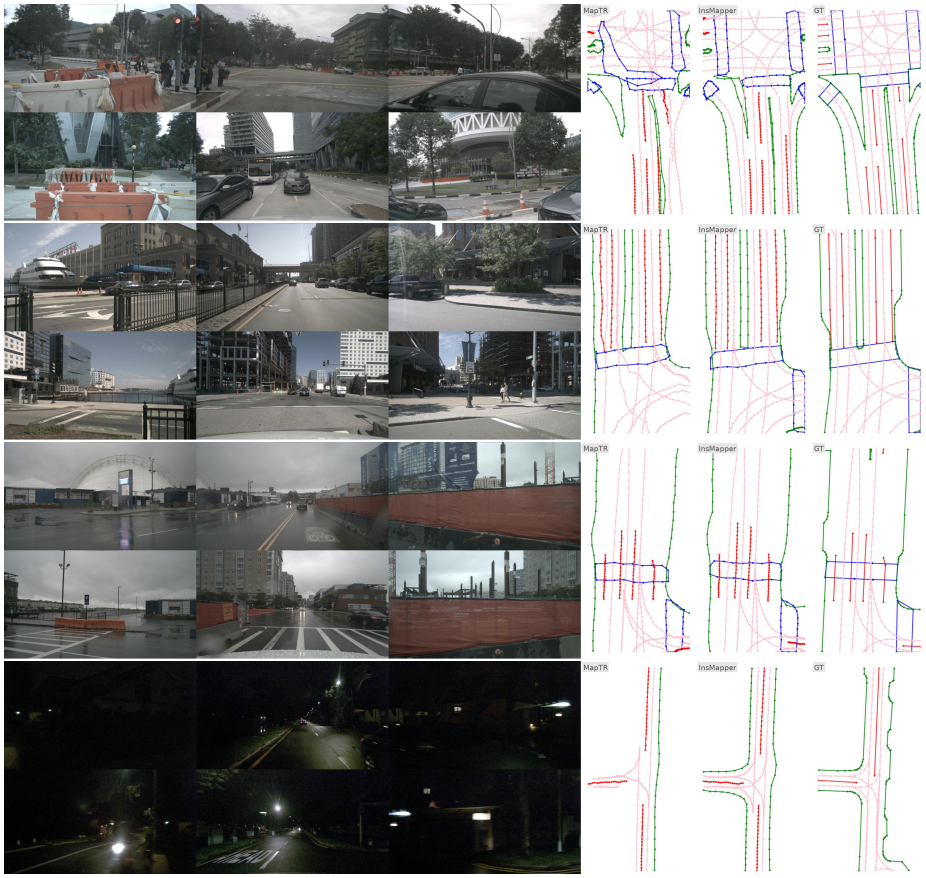


Fig. 7: Qualitative visualization. The predicted map contains four classes, i.e., road boundaries (green), lane splits (red), pedestrian crossing (blue), and lane centerlines (pink). InsMapper presents better detection results compared with the past work.

each query is summed with the mean of all queries of the same instance), fusion by the feed-forward network, and fusion by self-attention. The evaluation results are presented in Table 7. The outcomes of the query fusion methods vary significantly, with self-attention-based query fusion achieving the best results. Consequently, self-attention query fusion is incorporated into InsMapper.

Inner-instance feature aggregation. In InsMapper, the inner-instance self-attention module is incorporated into the decoder layers following the cross-attention module. We evaluate InsMapper under four different conditions: without the inner-instance self-attention module (no attention), replace the inner-instance self-attention module with a vanilla self-attention module (vanilla attention), without the random blocking (no random blocking, *i.e.*, ϵ=0),and with the module placed before the cross-attention (change position). The results are shown in Table 8. The outcomes reveal that removing or altering the proposed

Table 5: Ablation studies on key component designs. "QG", "QF" and "FA" represent hybrid query generation, inner-instance query fusion, and inner-instance feature aggregation, respectively.

QG	QF	FA	mAP	TOPO
			42.93	46.77
✓	✓		45.42(↑2.49)	49.19(↑2.42)
✓		✓	45.45(↑2.52)	48.54(↑1.77)
	✓	✓	46.93(↑4.00)	50.16(↑3.39)
✓	✓	✓	48.31(↑5.38)	51.58(↑4.81)

Table 6: Ablation studies on inner-instance query generation.

Query Scheme	mAP	TOPO
InsMapper-basic	46.62	49.39
InsMapper-hierarchical	46.93	50.16
InsMapper-hybrid	48.31	51.58

Table 7: Ablation studies on inner-instance query fusion.

Query Fusion	mAP	TOPO
No fusion	45.45	48.54
Mean	43.34	46.00
Feed-Forward	47.80	50.93
Self-attention	**48.31**	**51.58**

Table 8: Ablation studies on inner-instance feature aggregation. Multiple designs of the incorporated inner-instance self-attention layers are examined.

Method	mAP	TOPO
No attention	45.42	49.19
Vanilla attention	46.23	49.51
No random blocking	47.67	51.49
Change position	45.24	48.16
InsMapper	**48.31**	**51.58**

module can impair model performance. This observation confirms the effectiveness of the proposed module design.

6 Conclusion

In this paper, we introduced InsMapper, an end-to-end transformer-based model for on-the-fly vectorized HD map detection. InsMapper surpasses previous works by effectively leveraging inner-instance information to improve detection outcomes. Three exquisite designs have been proposed to utilize inner-instance information, including hybrid query generation, inner-instance query fusion, and inner-instance feature aggregation. The first two modules improve the quality of initialized line instances for detection, while the last module effectively refines the detected lines. The superiority of InsMapper is well exhibited through experiments on the challenging NuScenes and Argoverse 2 datasets. The evaluation results demonstrate the effectiveness and generality of InsMapper, making it a promising solution for the vectorized HD map detection task. In the future, InsMapper will be further enhanced for temporal-consistent vector map detection, making it more suitable for real-world applications.

Acknowledgement. This work is supported by the National Natural Science Foundation of China (No. 62201484), HKU Startup Fund, and HKU Seed Fund for Basic Research.

References

1. Bastani, F., He, S., Abbar, S., Alizadeh, M., Balakrishnan, H., Chawla, S., Madden, S., DeWitt, D.: Roadtracer: Automatic extraction of road networks from aerial images. In: CVPR (2018) 3
2. Caesar, H., Bankiti, V., Lang, A.H., Vora, S., Liong, V.E., Xu, Q., Krishnan, A., Pan, Y., Baldan, G., Beijbom, O.: nuscenes: A multimodal dataset for autonomous driving. In: CVPR (2020) 3, 10
3. Can, Y.B., Liniger, A., Paudel, D.P., Van Gool, L.: Structured bird's-eye-view traffic scene understanding from onboard images. In: ICCV (2021) 2, 4
4. Can, Y.B., Liniger, A., Paudel, D.P., Van Gool, L.: Topology preserving local road network estimation from single onboard camera image. In: CVPR (2022) 4
5. Carion, N., Massa, F., Synnaeve, G., Usunier, N., Kirillov, A., Zagoruyko, S.: End-to-end object detection with transformers. In: ECCV (2020) 2, 4, 7, 10
6. Da, F., Zhang, Y.: Path-aware graph attention for hd maps in motion prediction. In: ICRA (2022) 1
7. Ding, W., Qiao, L., Qiu, X., Zhang, C.: Pivotnet: Vectorized pivot learning for end-to-end hd map construction. In: ICCV (2023) 2, 3, 4, 5, 11
8. Girshick, R.: Fast r-cnn. In: ICCV (2015) 4
9. He, K., Chen, X., Xie, S., Li, Y., Dollár, P., Girshick, R.: Masked autoencoders are scalable vision learners. In: CVPR (2022) 9
10. He, K., Gkioxari, G., Dollár, P., Girshick, R.: Mask r-cnn. In: ICCV (2017) 4
11. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: CVPR (2016) 10
12. He, S., Balakrishnan, H.: Lane-level street map extraction from aerial imagery. In: WACV (2022) 4, 10
13. He, S., Bastani, F., Abbar, S., Alizadeh, M., Balakrishnan, H., Chawla, S., Madden, S.: Roadrunner: improving the precision of road network inference from gps trajectories. In: SIGSPATIAL (2018) 2, 10
14. He, S., Bastani, F., Jagwani, S., Alizadeh, M., Balakrishnan, H., Chawla, S., Elshrif, M.M., Madden, S., Sadeghi, M.A.: Sat2graph: road graph extraction through graph-tensor encoding. In: ECCV (2020) 3, 10
15. Homayounfar, N., Ma, W.C., Kowshika Lakshmikanth, S., Urtasun, R.: Hierarchical recurrent attention networks for structured online maps. In: CVPR (2018) 3
16. Homayounfar, N., Ma, W.C., Liang, J., Wu, X., Fan, J., Urtasun, R.: Dagmapper: Learning to map by discovering lane topology. In: ICCV (2019) 3
17. Lee, Y., Park, J.: Centermask: Real-time anchor-free instance segmentation. In: CVPR (2020) 11
18. Li, F., Zhang, H., Liu, S., Guo, J., Ni, L.M., Zhang, L.: Dn-detr: Accelerate detr training by introducing query denoising. In: CVPR (2022) 4
19. Li, F., Zhang, H., Xu, H., Liu, S., Zhang, L., Ni, L.M., Shum, H.Y.: Mask dino: Towards a unified transformer-based framework for object detection and segmentation. In: CVPR (2023) 4
20. Li, Q., Wang, Y., Wang, Y., Zhao, H.: Hdmapnet: A local semantic map learning and evaluation framework. In: ICRA (2022) 2, 4
21. Li, S., Yang, K., Shi, H., Zhang, J., Lin, J., Teng, Z., Li, Z.: Bi-mapper: Holistic bev semantic mapping for autonomous driving. RAL (2023) 4
22. Li, Z., Wang, W., Li, H., Xie, E., Sima, C., Lu, T., Qiao, Y., Dai, J.: Bevformer: Learning bird's-eye-view representation from multi-camera images via spatiotemporal transformers. In: ECCV (2022) 2, 4, 5

23. Liang, M., Yang, B., Hu, R., Chen, Y., Liao, R., Feng, S., Urtasun, R.: Learning lane graph representations for motion forecasting. In: ECCV (2020) 1, 2
24. Liao, B., Chen, S., Wang, X., Cheng, T., Zhang, Q., Liu, W., Huang, C.: Maptr: Structured modeling and learning for online vectorized hd map construction. In: ICLR (2023) 2, 3, 4, 5, 7
25. Liao, B., Chen, S., Zhang, Y., Jiang, B., Zhang, Q., Liu, W., Huang, C., Wang, X.: Maptrv2: An end-to-end framework for online vectorized hd map construction. arXiv:2308.05736 (2023) 2, 3, 4, 5, 7, 11, 12
26. Liu, T., hai Liao, Q., Gan, L., Ma, F., Cheng, J., Xie, X., Wang, Z., Chen, Y., Zhu, Y., Zhang, S., et al.: The role of the hercules autonomous vehicle during the covid-19 pandemic: An autonomous logistic vehicle for contactless goods transportation. IEEE Robotics & Automation Magazine (2021) 2
27. Liu, Y., Wang, Y., Wang, Y., Zhao, H.: Vectormapnet: End-to-end vectorized hd map learning. In: ICML (2023) 1, 2, 4
28. Liu, Z., Tang, H., Amini, A., Yang, X., Mao, H., Rus, D.L., Han, S.: Bevfusion: Multi-task multi-sensor fusion with unified bird's-eye view representation. In: ICRA (2023) 4
29. Mi, L., Zhao, H., Nash, C., Jin, X., Gao, J., Sun, C., Schmid, C., Shavit, N., Chai, Y., Anguelov, D.: Hdmapgen: A hierarchical graph generative model of high definition maps. In: CVPR (2021) 4
30. Ng, M.H., Radia, K., Chen, J., Wang, D., Gog, I., Gonzalez, J.E.: Bev-seg: Bird's eye view semantic segmentation using geometry and semantic point cloud. In: CVPRW (2020) 2, 4
31. Philion, J., Fidler, S.: Lift, splat, shoot: Encoding images from arbitrary camera rigs by implicitly unprojecting to 3d. In: ECCV (2020) 2, 4
32. Qiao, L., Ding, W., Qiu, X., Zhang, C.: End-to-end vectorized hd-map construction with piecewise bezier curve. In: CVPR (2023) 2, 4
33. Qiao, Z., Yu, Z., Yin, H., Shen, S.: Online monocular lane mapping using catmull-rom spline. In: IROS (2023) 4
34. Ross, S., Gordon, G., Bagnell, D.: A reduction of imitation learning and structured prediction to no-regret online learning. In: AISTATS (2011) 3
35. Shin, J., Rameau, F., Jeong, H., Kum, D.: Instagram: Instance-level graph modeling for vectorized hd map learning. In: CVPRW (2023) 2, 4
36. Wang, Y., Zhang, X., Yang, T., Sun, J.: Anchor detr: Query design for transformer-based detector. In: AAAI (2022) 4
37. Wilson, B., Qi, W., Agarwal, T., Lambert, J., Singh, J., Khandelwal, S., Pan, B., Kumar, R., Hartnett, A., Pontes, J.K., et al.: Argoverse 2: Next generation datasets for self-driving perception and forecasting. In: NeurIPS (2021) 3, 10, 12
38. Xie, Z., Pang, Z., Wang, Y.: Mv-map: Offboard hd-map generation with multi-view consistency. In: ICCV (2023) 2, 4
39. Xu, Z., Liu, Y., Gan, L., Sun, Y., Liu, M., Wang, L.: Rngdet: Road network graph detection by transformer in aerial images. TGRS (2022) 3
40. Xu, Z., Liu, Y., Sun, Y., Liu, M., Wang, L.: Centerlinedet: Road lane centerline graph detection with vehicle-mounted sensors by transformer for high-definition map creation. In: ICRA (2023) 2, 4
41. Xu, Z., Liu, Y., Sun, Y., Liu, M., Wang, L.: Rngdet++: Road network graph detection by transformer with instance segmentation and multi-scale features enhancement. RAL (2023) 2, 3, 10
42. Xu, Z., Sun, Y., Liu, M.: icurb: Imitation learning-based detection of road curbs using aerial images for autonomous driving. RAL (2021) 3

43. Xu, Z., Sun, Y., Liu, M.: Topo-boundary: A benchmark dataset on topological road-boundary detection using aerial images for autonomous driving. RAL (2021) 3
44. Yan, Y., Mao, Y., Li, B.: Second: Sparsely embedded convolutional detection. Sensors (2018) 11
45. Zhang, H., Li, F., Liu, S., Zhang, L., Su, H., Zhu, J., Ni, L.M., Shum, H.Y.: Dino: Detr with improved denoising anchor boxes for end-to-end object detection. In: ICLR (2023) 4
46. Zhu, X., Su, W., Lu, L., Li, B., Wang, X., Dai, J.: Deformable detr: Deformable transformers for end-to-end object detection. In: ICLR (2021) 2, 4, 5

Supplementary document of InsMapper

Zhenhua Xu, Kwan-Yee.K.Wong, and Hengshuang Zhao

The University of Hong Kong

1 Experiment

1.1 Experiment settings

In this paper, we employ a learning rate of 6×10^{-4} and a weight decay rate of 0.01. The experiments involve 100 instance queries ($N_I = 100$) and 20 point queries ($n_p = 20$), conducted using eight NVIDIA GeForce RTX 3090 GPUs, each equipped with 24GB of memory. For the ResNet backbones, we set the batch size to 4.

In our comparison experiments, we keep parameter settings of all methods exactly the same for fair comparison, such as batch size, network layers, number of input queries, input data resolution, etc.

1.2 Road elements

InsMapper detects four types of road elements essential for vectorized HD maps: road boundaries (polyline), lane splits (polyline), pedestrian crossings (polygon), and lane centerlines (polyline with topology). It should be noted that most past works either detect three types of road elements (i.e., road boundaries, lane splits and pedestrian crossings) [8–10] or only detect one road element (e.g., lane centerlines) [1, 14].

Currently, the detected HD map is an undirected graph to unify all elements. If directed vectorized HD maps are required for specific applications (e.g., centerlines need directions), InsMapper can be easily adapted by employing directed ground truth HD maps as labels to train the network.

1.3 Topology Evaluation metric

To provide a comprehensive evaluation, we report a topology-level evaluation score in this paper, namely the TOPO metric score [3–5, 12–15]. The TOPO metric has been used in several previous studies [3–5] to evaluate the correctness of lane and road-network graph topology. The TOPO metric first randomly samples vertices v_i^* from the ground truth graph G^*. It then finds corresponding matched vertices $\hat{v}_i$ in the predicted graph $\hat{G}$ based on the closest distance. Using v_i^* as a seed node, TOPO calculates a sub-graph $G_{v_i^*}^*$, where the distance between all vertices and v_i^* is smaller than a predetermined threshold. Similarly, we obtain the sub-graph $\hat{G}_{\hat{v}_i}$ by taking $\hat{v}_i$ as the seed node. Finally, we measure the graph similarity of the two sub-graphs using precision, recall, and F1-score. The TOPO metric is the mean similarity F1-score of all sampled vertex pairs $(v_i^*, \hat{v}_i)$.

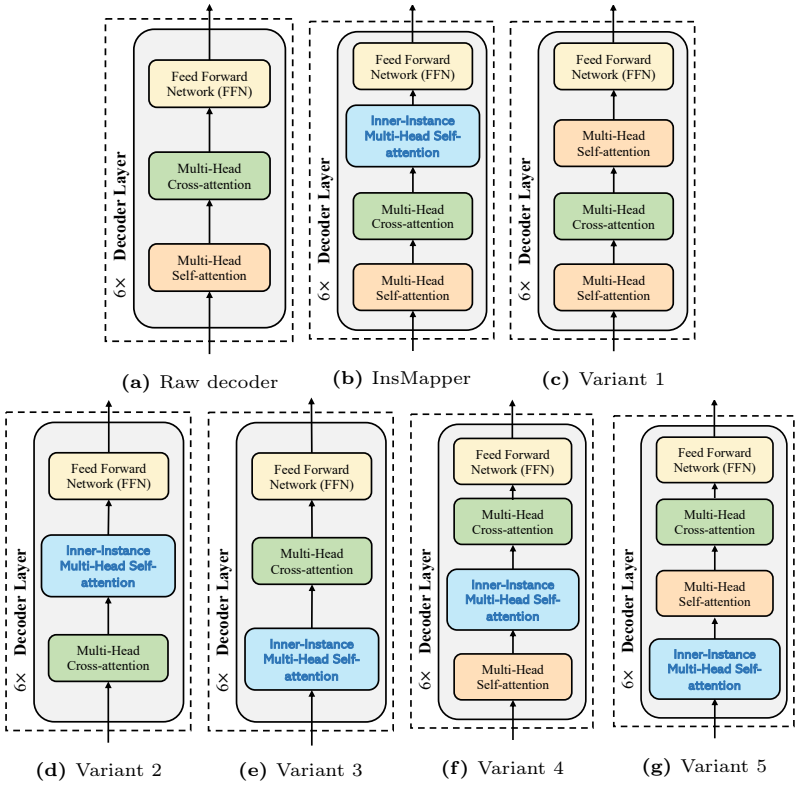


Fig. 1: Different transformer decoder designs. (a) Raw decoder of the deformable DETR. (b) InsMapper decoder. The proposed inner-instance self-attention module (blue) is incorporated. (c) Decoder variant 1. We replace the inner-instance self-attention module with a vanilla self-attention module. (d) Decoder variant 2. The self-attention module is removed from the InsMapper decoder. (e) Decoder variant 3. Swap the inner-instance self-attention module with the cross-attention module of the variant 2. (f) Variant 4. Place the inner-instance self-attention module before the cross-attention module. (g) Variant 5. Place the inner-instance self-attention module before the self-attention module. InsMapper decoder is the best design. Changing the structure of the proposed decoder will degrade the final performance.

2 Inner-instance feature aggregation

2.1 Ratio of random blocking

There is an ϵ possibility that the attention between inner-instance points is blocked (set to one, blocked). This blocking operation is inspired by [2] to enhance the robustness of the proposed module. ϵ controls the ratio of blocked attention masks. This concept is similar to dropout for better performance. However, dropout is applied after the *softmax* on the whole attention mask, while the blocking operation is before the *softmax* and it is operated only on the

Table 1: Quantitative results of ablation studies about the mask ratio.

ϵ	AP_{ped}	AP_{div}	AP_{bound}	AP_{center}	mAP	TOPO
0 (No masked attn)	42.48	50.66	53.22	41.54	46.98	51.49
25% (InsMapper)	**44.36**	**53.36**	52.77	42.35	**48.31**	51.58
50%	43.67	52.57	**53.98**	**42.95**	48.17	**51.59**
80%	42.27	52.49	53.64	41.34	47.41	49.20

Table 2: Quantitative results of ablation studies about transformer decoder designs.

Position	AP_{ped}	AP_{div}	AP_{bound}	AP_{center}	mAP	TOPO
Raw Decoder	39.41	50.04	52.13	40.10	45.42	49.19
Variant 1	40.50	50.78	52.17	41.46	46.23	49.51
Variant 2	40.96	47.85	51.11	40.45	45.09	47.98
Variant 3	40.92	46.40	51.94	38.91	44.54	46.90
Variant 4	39.09	50.09	51.56	40.21	45.24	48.16
Variant 5	44.04	49.59	52.18	41.78	46.90	49.52
InsMapper	**44.36**	**53.36**	**52.77**	**42.35**	**48.31**	**51.58**

inner-instance attention mask. Experimental results with different ϵ values are shown in Table 1.

2.2 Transformer Decoder Architectures

The transformer decoder of InsMapper may have multiple variants, which are visualized in Figure 1. There might be multiple kinds of attention layers and their positions can be altered. The evaluation results are reported in Table 2. Based on the results, the transformer decoder of InsMapper has the optimal design among these variants.

3 Instance-level class prediction

Although each instance contains n_p points, it should only have one predicted class for consistency. In MapTR, after obtaining point embeddings from the transformer decoder, it calculates a new instance-level embedding by taking the mean of all points in an instance. Then, the mean embedding is sent to the class head for class prediction. Differently, in InsMapper, we propose to use concatenation for class prediction, which preserves more information on points. Two class prediction methods are visualized

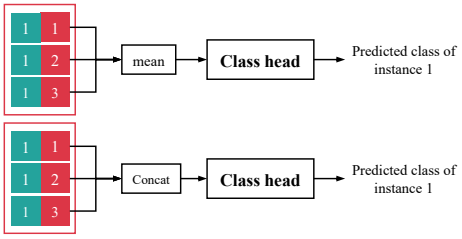


Fig. 2: Class head designs. MapTR leverages the mean of points for aggregation (upper). While InsMapper uses concatenation (lower).

in Figure 2. Experiment results are reported in Table 3. From the results, it is noted that concatenation achieves a slight performance gain. Thus, the concatenation method is used for class prediction in InsMapper.

Table 3: Quantitative results of ablation studies on the class head design.

Aggregation Method	AP_{ped}	AP_{div}	AP_{bound}	AP_{center}	mAP	TOPO
Mean	42.31	53.22	**53.70**	**43.16**	48.10	51.19
Concatenation (InsMapper)	**44.36**	**53.36**	52.77	42.35	**48.31**	**51.58**

4 Abandoned Designs

In our experiments, some designs are proved not effective for performance enhancement in our task. But we report them here which could be helpful for afterwards research.

4.1 Denosing DETR

In conventional object detection tasks, the denoising operation has been proven to effectively accelerate convergence and enhance overall performance [6, 7, 16]. However, this operation requires queries to be independent and identically distributed (i.i.d.). If this condition is not met, simply adding random noise to queries may not yield superior results. In our task, due to the strong correlation among inner-instance points, the denoising operation does not improve the vectorized HD map detection results. We report the outcomes of applying Dn-DETR in Table 4.

Table 4: Quantitative results of ablation studies about denoising DETR. For all metrics, a larger value indicates better performance.

Method	AP_{ped}	AP_{div}	AP_{bound}	AP_{center}	mAP	TOPO
InsMapper-Dn-DETR	43.74	51.34	**54.48**	42.33	47.97	50.37
InsMapper	**44.36**	**53.36**	52.77	**42.35**	**48.31**	**51.58**

In our experiments, we initially add random instance-level noise equally to all points within the same instance. Subsequently, we introduce random point-level noise to each point. Despite these modifications, we observe neither a significant improvement nor faster convergence. We attribute this unsatisfactory performance to the correlation between points.

4.2 Dynamic query generation

Another method to improve DETR in the query generation phase is dynamic query generation [2, 11, 16]. Unlike static query generation, where all queries are randomly initialized, dynamic queries are predicted based on the features

obtained by the transformer encoder. In other words, the output of the transformer encoder can be utilized to generate queries with better initialization.

After obtaining the transformer encoder output F, we create grids to partition F. Each grid $F_{x,y}$ contains the local information of the input image around coordinate (x, y). Then, each grid is used to predict a dynamic query, represented as a 4-D bounding box. In conventional detection tasks, objects are typically not very large, so each grid can make satisfactory predictions of dynamic queries. However, in our vectorized HD map detection task, target objects are often very thin and very long, which cannot be effectively predicted by local grids. Consequently, dynamic queries do not yield improvements. We present the results on dynamic query generation in Table 5.

Table 5: Quantitative results of ablation studies about dynamic queries.

Method	AP_{ped}	AP_{div}	AP_{bound}	AP_{center}	mAP	TOPO
InsMapper with dynamic query	43.14	48.86	52.17	38.65	45.71	43.83
InsMapper	**44.36**	**53.36**	**52.77**	**42.35**	**48.31**	**51.58**

4.3 Inter-instance self-attention

We also attempted to exploit the inter-instance information to further enhance the vectorized HD map detection results. The inter-instance self-attention module is incorporated into the decoder layers, similar to the inner-instance self-attention module, as shown in Figure 3. First, we predict the adjacency matrix, representing the connectivity of the predicted HD map. Some lane centerline instances may intersect with each other. Adjacent instances are illustrated by colored grids. Then, for adjacent instances, the corresponding attention mask grids are set to zero (attention is allowed). Otherwise, the attention is blocked. In this way, the information exchange between points in adjacent instances is allowed to better leverage the point correlations. However, this design significantly increases resource consumption while not noticeably improving the final results. The experiment results are shown in Table 6.

We believe the reason is the sparsity of the adjacency matrix. Under most circumstances, only a few instances intersect with each other, so the adjacency matrix is very sparse, providing limited additional inter-instance information. Furthermore, it may affect the inner-instance self-attention module, which is the main reason for the performance gains of InsMapper.

Therefore, at this stage, inter-instance self-attention is not used in InsMapper. But this could be an interesting topic for future exploration.

Table 6: Quantitative results of ablation studies about inter-instance self-attention.

Inter-instance Self-attn	AP_{ped}	AP_{div}	AP_{bound}	AP_{center}	mAP	TOPO	FPS
Yes	42.40	53.21	**54.03**	**43.03**	48.17	**52.63**	6.3
No (InsMapper)	**44.36**	**53.36**	52.77	42.35	**48.31**	51.58	**7.7**

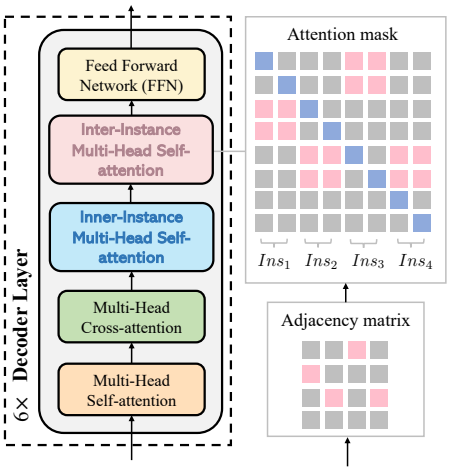


Fig. 3: Decoder of InsMapper with inter-instance self-attention module. In this module (the pink one), the attention between points of non-adjacent instances is blocked (grey grids). The attention is allowed for points of adjacent instances (pink grids), and diagonal grids (blue grids) to maintain the ego information of each point.

5 Additional qualitative visualizations

Qualitative visualizations on the Nuscenes validation set are shown in Figure 4 to Figure 7. The predicted map contains four classes, i.e., road boundaries (green), lane splits (red), pedestrian crossing (blue), and lane centerlines (pink). We visualize the vectorized HD map of the previous SOTA method MapTR, our proposed InsMapper, and the ground truth label. Models are trained with ResNet50 by 24 epochs.

References

1. Can, Y.B., Liniger, A., Paudel, D.P., Van Gool, L.: Structured bird's-eye-view traffic scene understanding from onboard images. In: ICCV (2021) 1
2. He, K., Chen, X., Xie, S., Li, Y., Dollár, P., Girshick, R.: Masked autoencoders are scalable vision learners. In: CVPR (2022) 2, 4
3. He, S., Balakrishnan, H.: Lane-level street map extraction from aerial imagery. In: WACV (2022) 1
4. He, S., Bastani, F., Abbar, S., Alizadeh, M., Balakrishnan, H., Chawla, S., Madden, S.: Roadrunner: improving the precision of road network inference from gps trajectories. In: SIGSPATIAL (2018) 1
5. He, S., Bastani, F., Jagwani, S., Alizadeh, M., Balakrishnan, H., Chawla, S., Elshrif, M.M., Madden, S., Sadeghi, M.A.: Sat2graph: road graph extraction through graph-tensor encoding. In: ECCV (2020) 1
6. Li, F., Zhang, H., Liu, S., Guo, J., Ni, L.M., Zhang, L.: Dn-detr: Accelerate detr training by introducing query denoising. In: CVPR (2022) 4
7. Li, F., Zhang, H., Xu, H., Liu, S., Zhang, L., Ni, L.M., Shum, H.Y.: Mask dino: Towards a unified transformer-based framework for object detection and segmentation. In: CVPR (2023) 4
8. Li, Q., Wang, Y., Wang, Y., Zhao, H.: Hdmapnet: A local semantic map learning and evaluation framework. In: ICRA (2022) 1
9. Liao, B., Chen, S., Wang, X., Cheng, T., Zhang, Q., Liu, W., Huang, C.: Maptr: Structured modeling and learning for online vectorized hd map construction. In: ICLR (2023) 1
10. Liu, Y., Wang, Y., Wang, Y., Zhao, H.: Vectormapnet: End-to-end vectorized hd map learning. In: ICML (2023) 1
11. Wang, Y., Zhang, X., Yang, T., Sun, J.: Anchor detr: Query design for transformer-based detector. In: AAAI (2022) 4
12. Xu, Z., Liu, Y., Gan, L., Hu, X., Sun, Y., Wang, L., Liu, M.: csboundary: City-scale road-boundary detection in aerial images for high-definition maps. arXiv:2111.06020 (2021) 1
13. Xu, Z., Liu, Y., Gan, L., Sun, Y., Liu, M., Wang, L.: Rngdet: Road network graph detection by transformer in aerial images. TGRS (2022) 1
14. Xu, Z., Liu, Y., Sun, Y., Liu, M., Wang, L.: Centerlinedet: Road lane centerline graph detection with vehicle-mounted sensors by transformer for high-definition map creation. In: ICRA (2023) 1
15. Xu, Z., Liu, Y., Sun, Y., Liu, M., Wang, L.: Rngdet++: Road network graph detection by transformer with instance segmentation and multi-scale features enhancement. RAL (2023) 1
16. Zhang, H., Li, F., Liu, S., Zhang, L., Su, H., Zhu, J., Ni, L.M., Shum, H.Y.: Dino: Detr with improved denoising anchor boxes for end-to-end object detection. In: ICLR (2023) 4

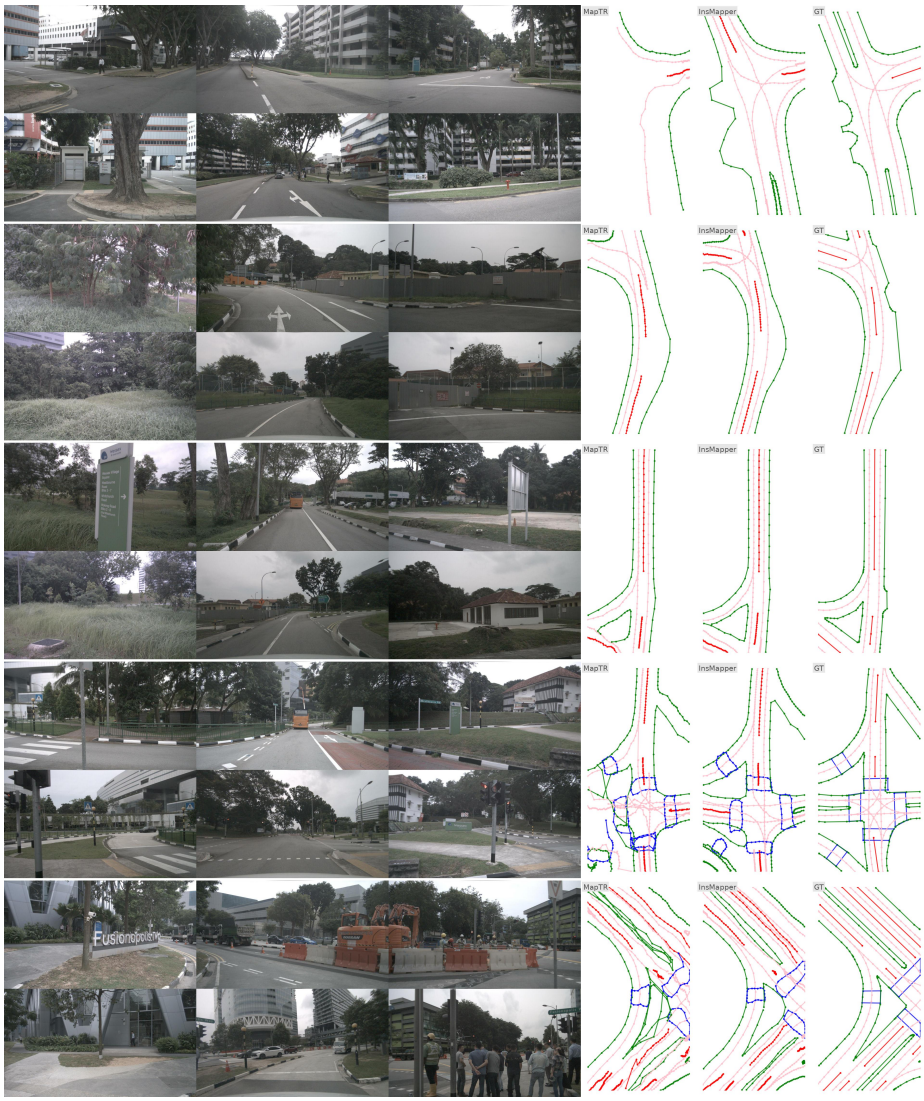


Fig. 4: Qualitative visualization. Left three columns are input 6 RGB camera images. For map columns, the first column presents MapTR's results, the second column features InsMapper's outcomes, and the final column depicts the ground truth map. The predicted map contains four classes, i.e., road boundaries (green), lane splits (red), pedestrian crossing (blue), and lane centerlines (pink).

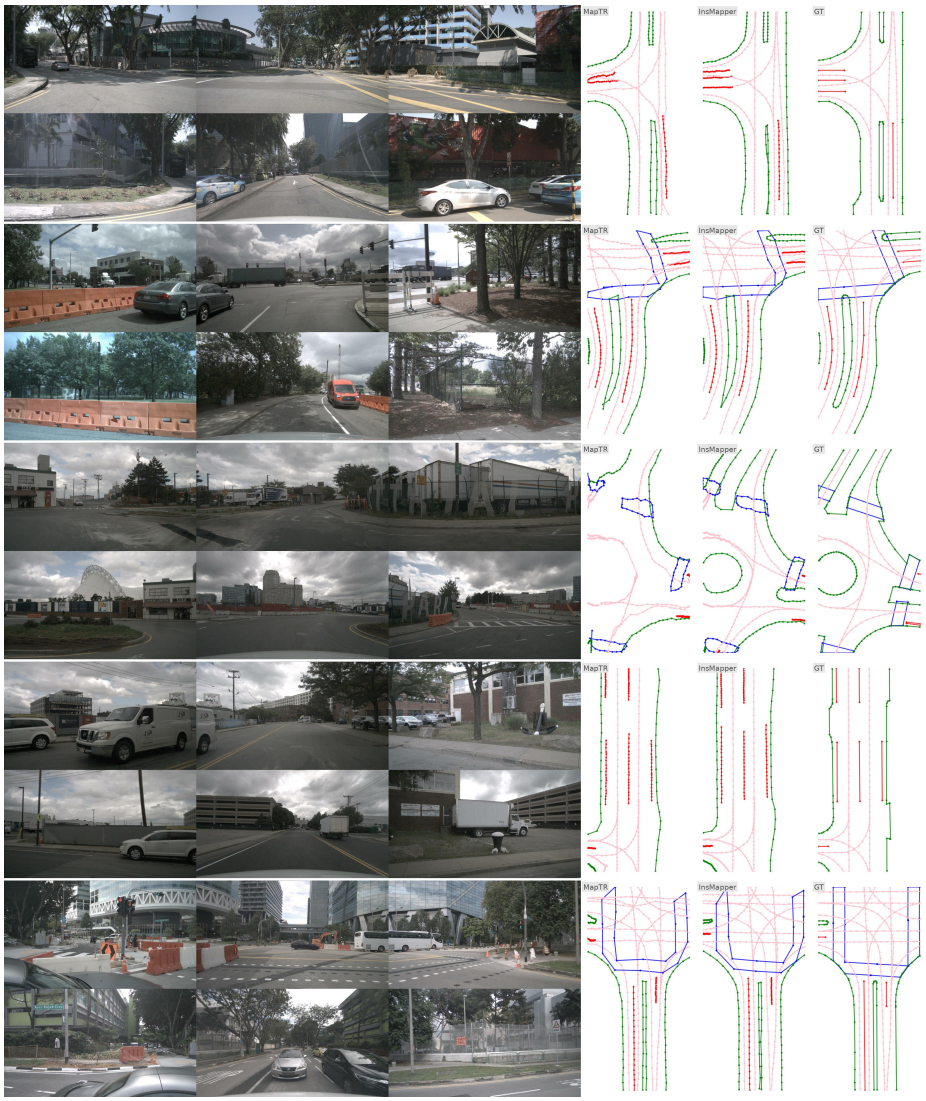


Fig. 5: Qualitative visualization. Left three columns are input 6 RGB camera images. For map columns, the first column presents MapTR's results, the second column features InsMapper's outcomes, and the final column depicts the ground truth map. The predicted map contains four classes, i.e., road boundaries (green), lane splits (red), pedestrian crossing (blue), and lane centerlines (pink).

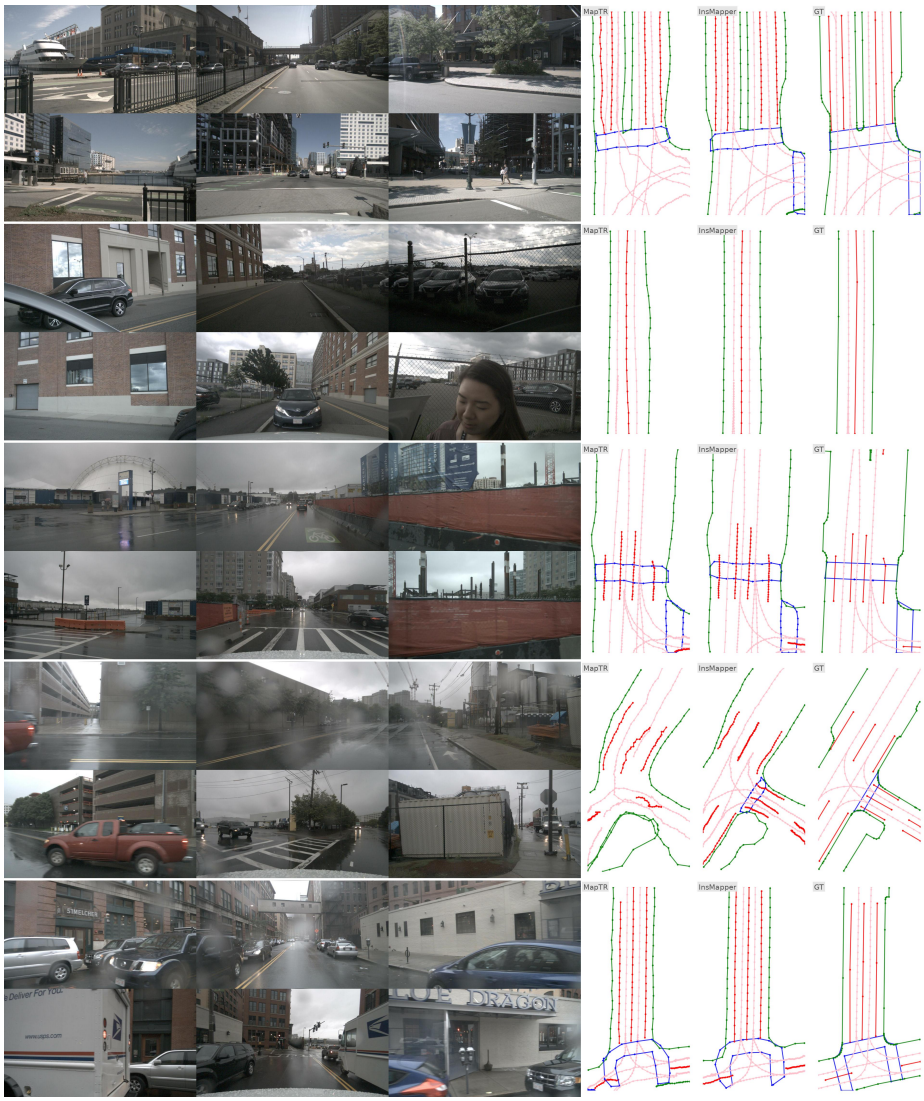


Fig. 6: Qualitative visualization. Left three columns are input 6 RGB camera images. For map columns, the first column presents MapTR's results, the second column features InsMapper's outcomes, and the final column depicts the ground truth map. The predicted map contains four classes, i.e., road boundaries (green), lane splits (red), pedestrian crossing (blue), and lane centerlines (pink).

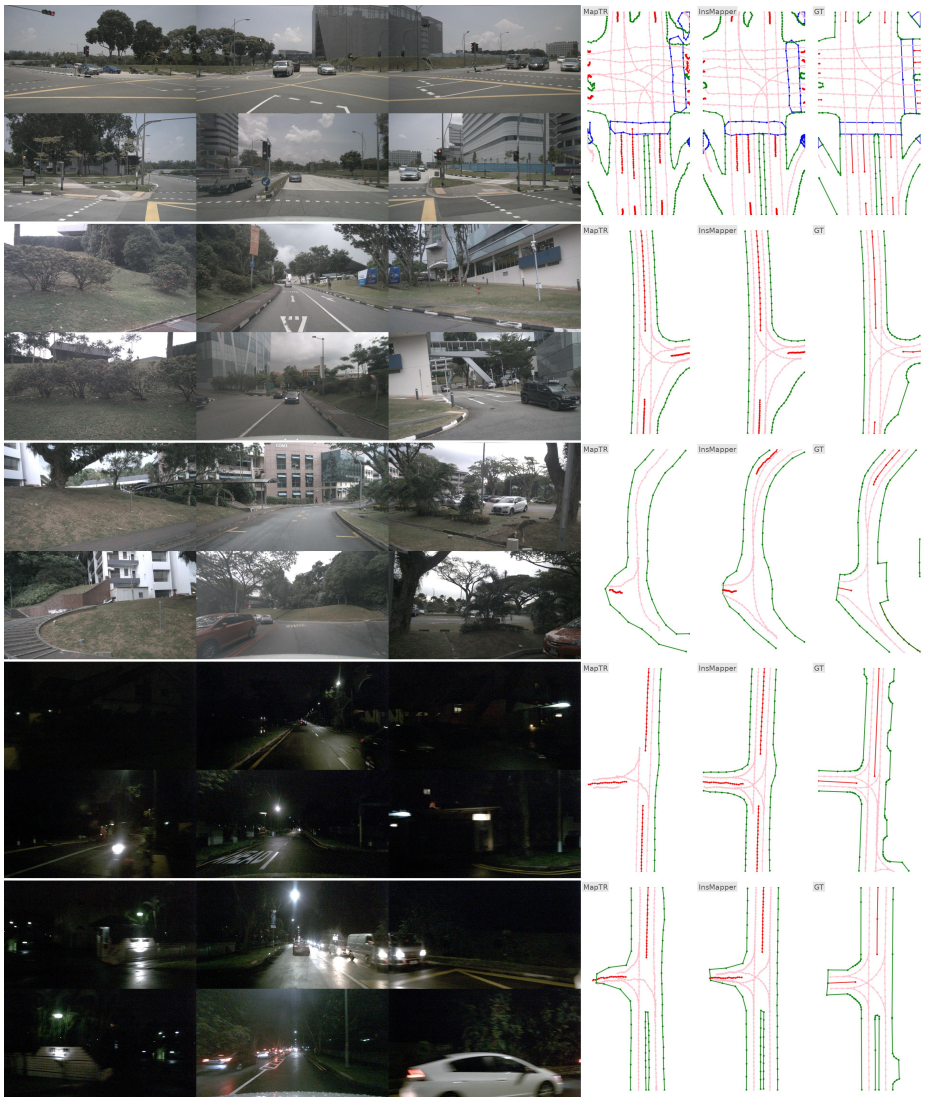


Fig. 7: Qualitative visualization. Left three columns are input 6 RGB camera images. For map columns, the first column presents MapTR's results, the second column features InsMapper's outcomes, and the final column depicts the ground truth map. The predicted map contains four classes, i.e., road boundaries (green), lane splits (red), pedestrian crossing (blue), and lane centerlines (pink).