MTR++: Multi-Agent Motion Prediction With Symmetric Scene Modeling and Guided Intention Querying

Shaoshuai Shi, Li Jiang, Dengxin Dai, and Bernt Schiele, *Fellow, IEEE*

***Abstract*—Motion prediction is crucial for autonomous driving systems to understand complex driving scenarios and make informed decisions. However, this task is challenging due to the diverse behaviors of traffic participants and complex environmental contexts. In this paper, we propose Motion TRansformer (MTR) frameworks to address these challenges. The initial MTR framework utilizes a transformer encoder-decoder structure with learnable intention queries, enabling efficient and accurate prediction of future trajectories. By customizing intention queries for distinct motion modalities, MTR improves multimodal motion prediction while reducing reliance on dense goal candidates. The framework comprises two essential processes: global intention localization, identifying the agent's intent to enhance overall efficiency, and local movement refinement, adaptively refining predicted trajectories for improved accuracy. Moreover, we introduce an advanced MTR++ framework, extending the capability of MTR to simultaneously predict multimodal motion for multiple agents. MTR++ incorporates symmetric context modeling and mutually-guided intention querying modules to facilitate future behavior interaction among multiple agents, resulting in scene-compliant future trajectories. Extensive experimental results demonstrate that the MTR framework achieves state-of-the-art performance on the highly-competitive motion prediction benchmarks, while the MTR++ framework surpasses its precursor, exhibiting enhanced performance and efficiency in predicting accurate multimodal future trajectories for multiple agents.**

***Index Terms*—Motion prediction, transformer, intention query, autonomous driving.**

I. Introduction

MOTION prediction constitutes a pivotal undertaking within the realm of contemporary autonomous driving systems, and it has gained significant attention in recent years due to its vital role in enabling robotic vehicles to understand driving scenarios and make judicious decisions [21], [24], [25], [26], [33], [39], [49], [63], [69]. The core of motion prediction lies in accurately anticipating the future actions of traffic participants by considering observed agent states and complex road maps. However, this task is challenging due to the inherent multimodal behaviors exhibited by agents and the intricacies of the surrounding environment.

To tackle these formidable challenges, prior studies [17], [31], [40] have delved into diverse strategies aimed at encoding the complex scene context. Some works [39], [50] have employed the encoded agent features in motion decoders based on Multi-Layer Perceptrons (MLPs) to directly predict multiple potential future trajectories for the agent. Nonetheless, these methodologies generally exhibit a bias towards predicting the most frequently occurring modes observed in the training data, thereby yielding homogeneous trajectories that inadequately capture the agent's multimodal behaviors. To improve trajectory predictions encompassing all potential future behaviors for the agent, alternative approaches [21], [67] have explored a goal-based strategy. This strategy involves using a dense set of goal candidates representing feasible destinations for the agent. By predicting the probability associated with each candidate being the actual destination, these methods generate a full trajectory for each selected goal candidate. While this strategy reduces trajectory uncertainty during model optimization, the performance of such methods is highly dependent on the density of goal candidates. Fewer candidates lead to decreased performance, while an excessive number of candidates significantly increases computation and memory costs. To address these challenges and enhance multimodal motion prediction while reducing reliance on dense goal candidates, we propose a novel collection of frameworks called Motion TRansformer (MTR) frameworks. These frameworks consist of an initial MTR framework and an advanced MTR++ framework.

In the MTR frameworks, we introduce a novel set of learnable intention queries integrated into a transformer encoder-decoder structure, which facilitates efficient motion prediction by employing each intention query to encompass the behavior prediction of a bunch of potential trajectories directed towards the same region. Guided by these intention queries, the MTR frameworks optimize two key tasks simultaneously. The first task is global intention localization, which aims to roughly identify the agent's intention, thereby enhancing overall efficiency. The second task is local movement refinement, which strives to adaptively refine the predicted trajectory for each intention, thereby improving accuracy. The proposed MTR frameworks not only foster a stable training process without depending on dense goal candidates

Manuscript received 31 May 2023; revised 15 October 2023; accepted 29 December 2023. Date of publication 12 January 2024; date of current version 3 April 2024. This work was supported by Max Planck Institute for Informatics. Recommended for acceptance by J. Gall. *(Shaoshuai Shi and Li Jiang contributed equally to this work.) (Corresponding author: Li Jiang.)*

The authors are with the Max Planck Institute for Informatics, Saarland Informatics Campus, 66123 Saarbrücken, Germany (e-mail: shaoshuaics@gmail.com; jlnemuru@gmail.com; ddai@mpi-inf.mpg.de; schiele@mpi-inf.mpg.de).

Digital Object Identifier 10.1109/TPAMI.2024.3352811

but also enable flexible and adaptable prediction by facilitating local refinement for each motion mode.

Specifically, the MTR frameworks introduce distinct learnable intention queries to handle trajectory prediction across different motion modes. To accomplish this, a limited number of spatially distributed intention points (e.g., 64 in our case) are initially generated for each category. These intention points effectively reduce uncertainty in future trajectories by encompassing both motion direction and velocity. Each intention query represents the learnable position embedding of a specific intention point, assuming responsibility for predicting the future trajectory of that particular motion mode. This approach not only enhances multimodal motion prediction by explicitly leveraging different queries for different modes but also eliminates the necessity of dense goal candidates, as each query assumes responsibility for a substantial destination region. Moreover, the MTR frameworks employ the classification probability of all intention queries to roughly localize the agent's motion intention, while the predicted trajectory of each intention query undergoes iterative refinement through stacked transformer decoder layers. This iterative refinement process involves continually retrieving fine-grained local features of each trajectory. Our experiments show that these two complementary processes have demonstrated remarkable efficacy in predicting multimodal future motion.

In contrast to the initial MTR framework presented in our previous version [46], which focuses on the multimodal motion prediction of a single agent, we introduce an advanced MTR++ framework that extends the capability to predict multimodal motion concurrently for multiple agents (see Fig. 1). Instead of individually encoding the scene context around each agent as in the previous version, we propose a novel symmetric scene context modeling strategy. This strategy employs a shared context encoder to symmetrically encode the entire scene for each agent, incorporating a novel query-centric self-attention module to jointly capture the intricate scene context information within their respective local coordinate systems. Furthermore, we introduce mutually-guided intention quering module in the motion decoder network, enabling agents to interact and influence each other's behavior. This facilitates more precise and scene-compliant joint motion prediction for multiple agents. Through these two enhancements, experimental results demonstrate that compared to the initial MTR framework, the MTR++ framework effectively predicts more accurate multimodal future trajectories for multiple agents simultaneously. Additionally, as shown in Fig. 1, the efficiency advantage of the MTR++ framework becomes more pronounced as the number of agents increases.

Our contributions are four-fold: (1) We introduce the MTR frameworks, which incorporate a novel set of learnable intention queries within the transformer encoder-decoder architecture for motion prediction. By customizing intention queries to address distinct motion modalities, the MTR frameworks not only achieve more precise multimodal future trajectory predictions that encompass a wide range of possibilities but also obviate reliance on dense goal candidates. (2) We propose the advanced MTR++ framework for simultaneous multimodal motion prediction of multiple agents. This framework incorporates two key components: a symmetric scene context modeling module that allows for shared context encoding among multiple agents, and a mutually-guided intention querying module that facilitates the interaction of agents' future behaviors and enables the prediction of scene-compliant trajectories. (3) The initial MTR framework achieves state-of-the-art performance on the motion prediction benchmark of Waymo Open Motion Dataset (WOMD) [15], surpassing previous ensemble-free approaches with a remarkable mAP gain of +8.48%. Additionally, the MTR++ framework further enhances the capabilities of the initial MTR framework, enabling concurrent joint multimodal motion prediction for multiple agents and improving both performance and efficiency. (4) Notably, our initial MTR and MTR++ frameworks won the championship of the highly-competitive Waymo Motion Prediction Challenge in 2022 [57] and 2023 [57], respectively, demonstrating their superiority and effectiveness.

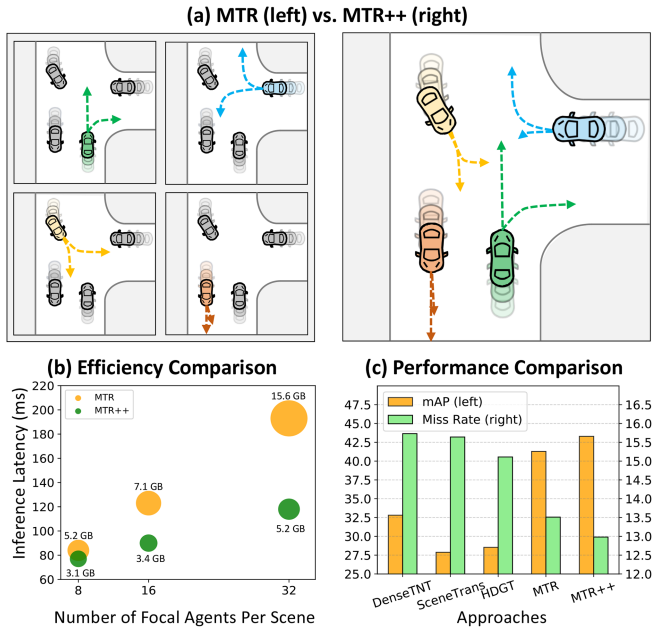


Fig. 1. Comparison of MTR and MTR++ frameworks. The MTR++ framework surpasses its predecessor, MTR, in several key aspects. In subfigure (a), MTR++ demonstrates its ability to predict the future trajectories of multiple agents simultaneously. Notably, in subfigure (b), MTR++ excels in both inference speed and memory efficiency, particularly when dealing with a larger number of interested agents. Additionally, as depicted in subfigure (c), the MTR++ framework outperforms both MTR and all other approaches, achieving superior performance overall.

II. Related Work

Scene Context Encoding for Motion Prediction: The motion prediction task in autonomous driving scenarios involves the encoding of the input road map and agent history states to generate future trajectories of the agent, which plays a crucial role in this task. Prior works [4], [6], [9], [13], [35], [40], [66] have commonly employed rasterization techniques to convert the scene context into images, allowing for processing with convolutional neural networks (CNNs). LaneGCN [31] utilizes

a lane graph to capture the topological information of the map, and recent works [21], [39], [47], [50] have widely adopted VectorNet [17] for its efficiency and scalability. VectorNet represents both road maps and agent trajectories as polylines. In our MTR frameworks, we also adopt this vector representation. However, instead of constructing a global graph of polylines, we advocate employing a transformer encoder on a locally connected graph. This strategy not only better preserves the input's locality structure but also improves memory efficiency, enabling larger map encodings for long-term motion prediction.

Multimodal Future Behavior Modeling: Given the encoded scene context features, existing works explore diverse strategies for modeling the agent's multimodal future behaviors. Early works [2], [22], [43], [44], [48] suggests generating a set of trajectory samples to approximate the output distribution. other studies [10], [23], [37], [41], [45] parameterize multimodal predictions with Gaussian Mixture Models (GMMs) to generate compact distribution. The HOME series [18], [19] generates trajectories by sampling a predicted heatmap. IntentNet [8] considers intention prediction as a classification problem involving eight high-level actions, while [33] proposes a region-based training strategy. Goal-based methods [16], [34], [44], [67] represent another category, estimating several agent goal points before completing the full trajectory for each goal.

The large-scale Waymo Open Motion Dataset (WOMD) [15] has recently been introduced for long-term motion prediction. To address this challenge, DenseTNT [21] employs a goal-based strategy to classify trajectory endpoints from dense goal candidates. Other works directly predict the future trajectories based on the encoded agent features [39] or latent anchor embedding [50]. Nonetheless, the goal-based strategy raises efficiency concerns due to the numerous goal candidates, while the direct-regression strategy converges slowly and exhibits a bias to predict homogeneous trajectories since various motion modes are regressed from identical agent features. In contrast, our MTR frameworks employ a small set of learnable intention queries to address these limitations, facilitating the generation of future trajectories with extensive modalities and eliminating numerous goal candidates by employing mode-specific learnable intention queries to predict different motion modes.

Furthermore, we employ Gaussian Mixture Models in tandem with intention queries to model the continuous distribution of an agent's multimodal future behavior at each time step. This approach yields a parametric multimodal future distribution as the output, capable of generating occurrence probabilities for any specified future trajectory. In contrast, utilizing a set of sparse trajectories fails to provide a continuous and compact future distribution. Similarly, predicting dense future heatmaps incurs a substantial computational cost, necessitating a compromise between resolution and computational efficiency.

Simultaneous Motion Prediction of Multiple Agents: In predicting an individual agent's future trajectories, state-of-the-art works [21], [50], including our previous version [46], typically customize the scene context encoding for that agent by normalizing all inputs centered on it. This strategy results in computational inefficiencies when predicting motion for multiple agents. To simultaneously predict future trajectories for multiple agents, SceneTransformer [39] encodes all road graphs and agents into a scene-centric embedding applicable to all agents. However, their feature encoding still relies on a global coordinate system centered on an agent of interest (e.g., the autonomous vehicle), limiting its performance for off-center agents. Recent works [27], [69] explore encoding the agents' node features in an ego-centric coordinate system, while they generally construct hand-crafted relation graphs and alternate node-edge updating strategy. In contrast, our MTR++ framework introduces symmetric scene context modeling for all agents with innovative query-centric self-attention, operating on a straightforward polyline graph using the native transformer encoder module with relative position encoding, thereby promoting more efficient and concise shared scene context encoding.

To enable the behavioral interaction of multiple agents, recent research M2I [47] introduces a triad of models, initially employing a relation predictor to categorize two interacting agents as influencer and reactor, followed by the sequential generation of their future trajectories via a marginal predictor and a conditional predictor, respectively. Conversely, our MTR++ framework integrates mutually-guided intention queries, fostering the behavioral interaction of more than two agents within a unified model, wherein their predicted future behaviors naturally interact through stacked transformer decoder layers, thereby yielding superior scene-compliant trajectories with higher efficiency for multiple agents.

Transformer: Transformer [51] has been extensively employed in natural language processing [3], [12] and computer vision [5], [14], [52], [53], [54], [64]. Our approach draws inspiration from DETR [5] and its subsequent works [11], [29], [32], [36], [62], [65], [70], particularly DAB-DETR [32], where the object query serves as the positional embedding of an anchor box. Motivated by their notable success in object detection, we introduce the innovative concept of learnable intention query to model multimodal motion prediction with prior intention points. Each intention query is tasked with predicting a specific motion mode and enables iterative motion refinement by integrating with transformer decoders.

III. MTR for Multimodal Motion Prediction

We propose Motion TRansformer (MTR), which adopts a novel transformer encoder-decoder architecture incorporating iterative motion refinement for predicting multimodal future motion. The overall framework is presented in Fig. 2. In Section III-A, we introduce our encoder network for scene context modeling. In Section III-B, we present motion decoder network with a novel concept of intention query for predicting multimodal trajectories. Finally, in Section III-C, we introduce the optimization process of our framework.

A. Transformer Encoder for Scene Context Modeling

The forthcoming actions of the agents are greatly influenced by their interactions and the road map. To incorporate this contextual information into the model, prior approaches have employed diverse techniques, such as constructing a comprehensive interactive graph [17], [21] or condensing map features

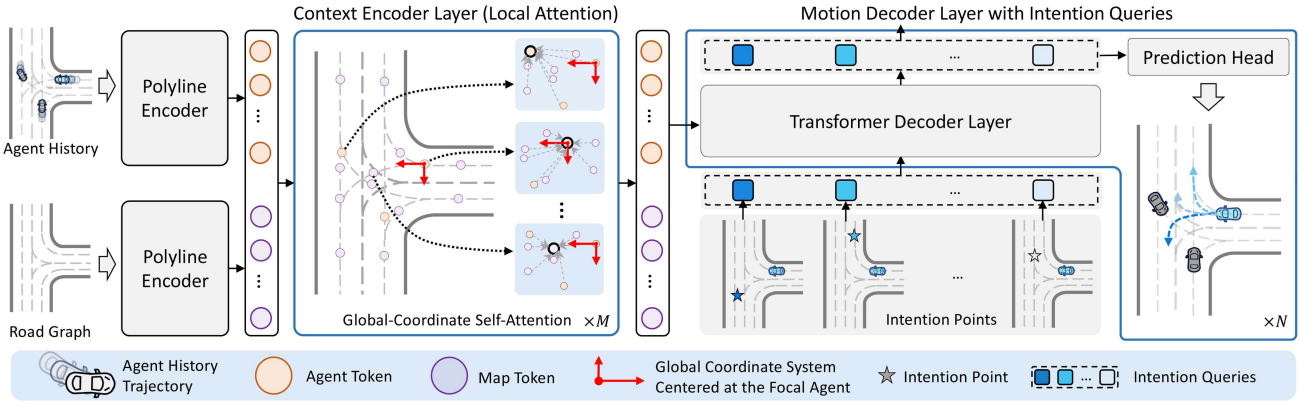


Fig. 2. Architecture of the MTR framework. In this framework, we first utilize two polyline encoders to encode the polylines derived from agent history trajectories and road lanes into token features. Next, multiple transformer encoder layers with local self-attention are utilized to model the relationships among different tokens within the global coordinate system centered around the focal agent of interest. This allows for a comprehensive understanding of the scene contextual information. Finally, a small set of learnable intention queries are integrated into the stacked transformer decoder layers to aggregate information from the encoded context features. Each intention query is responsible for predicting future trajectories towards a specific intention point, enabling the generation of multimodal future trajectories for the focal agent.

into agent-specific features [39], [50]. However, we argue that preserving the locality structure of the scene context, particularly the road map, is crucial. Thus, we introduce a transformer encoder network that utilizes local self-attention to better capture this structural information.

Input Representation With Single Focal Agent: We adopt the vectorized depiction [17] to arrange both input trajectories and road maps as polylines. When forecasting the motion of an individual focal agent, we employ the *focal-agent-centric* approach [21], [50], [67], which normalizes all inputs to the global coordinate system centered on this agent.

Concretely, the past states of N_a agents are denoted as $S_{\text{A}}^{(\text{g})} \in \mathbb{R}^{N_a \times T_h \times C_a}$ (where "g" indicating the global reference frame). Here, T_h represents the duration of the historical observations, and C_a corresponds to the dimensionality of the state information, encompassing factors such as position, orientation, and velocity. Zero-padding is applied to the positions of absent frames in trajectories comprising fewer than T_h frames. The road map is represented as $S_{\text{M}}^{(\text{g})} \in \mathbb{R}^{N_m \times n \times C_m}$, where N_m indicates the number of map polylines, n represents the number of points in each polyline, and C_m signifies the number of attributes for each point (e.g., location and road type). Both $S_{\text{A}}^{(\text{g})}$ and $S_{\text{M}}^{(\text{g})}$ are encoded utilizing a PointNet-like [42] polyline encoder as

$$F_{\text{A}}^{(\text{g})} = \phi\left(\text{MLP}\left(S_{\text{A}}^{(\text{g})}\right)\right), \qquad F_{\text{M}}^{(\text{g})} = \phi\left(\text{MLP}\left(S_{\text{M}}^{(\text{g})}\right)\right), \quad (1)$$

where $\text{MLP}(\cdot)$ represents a multi-layer perceptron, while ϕ denotes max-pooling, employed to encapsulate each polyline's feature as agent features $F_{\text{A}}^{(\text{g})} \in \mathbb{R}^{N_a \times D}$ and map features $F_{\text{M}}^{(\text{g})} \in \mathbb{R}^{N_m \times D}$ with a feature dimension of D.

These two types of polyline features are concatenated to form the following input token features, denoted as $F_{\text{AM}}^{(\text{g})} = [F_{\text{A}}^{(\text{g})}, F_{\text{M}}^{(\text{g})}] \in \mathbb{R}^{(N_a+N_m)\times D}$. The positions of these tokens are denoted as $P_{\text{AM}}^{(\text{g})} = [P_{\text{A}}^{(\text{g})}, P_{\text{M}}^{(\text{g})}] \in \mathbb{R}^{(N_a+N_m)\times 2}$, where we utilize the most recent positions for agent tokens (denoted as $P_{\text{A}}^{(\text{g})} \in \mathbb{R}^{N_a \times 2}$) and polyline centers for map tokens (denoted as $P_{\text{M}}^{(\text{g})} \in \mathbb{R}^{N_m \times 2}$).

Scene Context Encoding With Local Transformer Encoder: The local structure of scene context is vital for motion prediction. For instance, the relationship between two parallel lanes is essential for modeling lane-changing behavior, but utilizing attention on a globally connected graph treats all lane relations equally. Therefore, we incorporate prior knowledge into the context encoder by employing local attention, which better preserves the locality structure and is more memory-efficient. Specifically, the attention module of each transformer encoder layer can be expressed as

$$\begin{aligned} F'^{(\text{g})}_{\text{AM}}[i] = \text{MHSA}\big(&\text{Q: } [F^{(\text{g})}_{\text{AM}}[i], \text{PE}\Big(P^{(\text{g})}_{\text{AM}}[i]\Big)], \\ &\text{K: } \Big\{\Big[F^{(\text{g})}_{\text{AM}}[j], \text{PE}\Big(P^{(\text{g})}_{\text{AM}}[j]\Big)\Big]\Big\}_{j\in\Omega(i)}, \\ &\text{V: } \Big\{F^{(\text{g})}_{\text{AM}}[j]\Big\}_{j\in\Omega(i)}\big), \end{aligned} \quad (2)$$

where $i \in \{1, \ldots, N_a + N_m\}$. $\Omega(i)$ indicates the index set of the k neighborhoods of ith token. $\text{MHSA}(\cdot_{\text{query}}, \cdot_{\text{key}}, \cdot_{\text{value}})$ denotes multi-head self-attention layer [51]. $\text{PE}(\cdot)$ signifies the sinusoidal positional encoding of input tokens. $F'^{(\text{g})}_{\text{AM}}[i] \in \mathbb{R}^D$ is the output feature of the ith token of this encoder layer. Thanks to this local self-attention, our framework can encode a considerably larger scene context.

By stacking multiple transformer encoder layers, the encoder network generates the token features $F'^{(\text{g})}_{\text{AM}} \in \mathbb{R}^{(N_a+N_m)\times 2}$. We decompose these features to obtain the agent history features $F_{\text{A}}^{(\text{g, past})} \in \mathbb{R}^{N_a \times D}$ and map features $F_{\text{M}}^{(\text{g})} \in \mathbb{R}^{N_m \times D}$, where the agent history features will be further enhanced as $F_{\text{A}}^{(\text{g})} \in \mathbb{R}^{N_a \times D}$ by the following dense future prediction module. Note that in the following sections, we employ the same notations for convenience, referring to $F_{\text{A}}^{(\text{g})} \in \mathbb{R}^{N_a \times D}$ and $F_{\text{M}}^{(\text{g})} \in \mathbb{R}^{N_m \times D}$ to represent the agent features and map features, respectively, which are encoded by the context encoder.

Dense Future Prediction for All Agents: Interactions with other agents significantly influence the behaviors of our

focal agent. Existing methods, such as hub-host networks [71], dynamic relational reasoning [30], and social spatial-temporal networks [61], mainly focus on learning past interactions but often overlook future trajectory interactions. To compensate for this limitation, we propose a method that densely predicts future states for all agents using a straightforward regression head on the encoded history features $F_{\text{A}}^{(\text{g, past})}$, as follows

$$S_{\text{A}}^{(\text{g, future})} = \text{MLP}\left(F_{\text{A}}^{(\text{g, past})}\right), \tag{3}$$

where $S_{\text{A}}^{(\text{g, future})} \in \mathbb{R}^{N_a \times (T_f \times 4)}$ includes the future position and velocity of each agent, and T_f denotes the number of future frames to be predicted. The predicted trajectories $S_{\text{A}}^{(\text{g, future})}$ are encoded using the same polyline encoder as in (1), producing features $F_{\text{A}}^{(\text{g, future})} \in \mathbb{R}^{N_a \times D}$. These features are combined with $F_{\text{A}}^{(\text{g, past})} \in \mathbb{R}^{N_a \times D}$ using feature concatenation and three MLP layers, resulting in enhanced features $F_{\text{A}}^{(\text{g})} \in \mathbb{R}^{N_a \times D}$.

By supplying the motion decoder network with additional future context information, this approach effectively improves the model's capability to predict more accurate future trajectories for the focal agent. Experimental results demonstrate that this simple auxiliary task effectively enhances the performance of multimodal motion prediction.

B. Motion Decoder With Intention Query

To facilitate multimodal motion prediction, the MTR framework utilizes a transformer-based motion decoder network that incorporates the previously encoded scene context features. We introduce the concept of intention query, which facilitates multimodal motion prediction through the joint optimization of global intention localization and local movement refinement. As depicted in Fig. 2, the motion decoder network consists of stacked transformer decoder layers that iteratively refine predicted trajectories utilizing learnable intention queries. Next, we elaborate on the detailed structure.

Learnable Intention Query:[1] To efficiently and precisely pinpoint an agent's potential motion intentions, we propose the learnable *intention query* to diminish the uncertainty of future trajectories by employing different intention queries for different motion modes. Specifically, for each category, we generate $\mathcal{K}$ representative intention points $I^{(\text{s})} \in \mathbb{R}^{\mathcal{K} \times 2}$ (where "s" indicating a single focal agent) by utilizing the k-means clustering algorithm on the endpoints of ground-truth (GT) trajectories in the training dataset (refer to Fig. 3). Each intention point embodies an implicit motion mode, accounting for both motion direction and velocity. Given the intention points of a single focal agent, we model each intention query as the learnable positional embedding of a specific intention point

$$E_{\text{I}}^{(\text{s})}[i] = \text{MLP}\left(\text{PE}\left(I^{(\text{s})}[i]\right)\right), \tag{4}$$

where $i \in \{1, \ldots, \mathcal{K}\}$ and $E_{\text{I}}^{(\text{s})} \in \mathbb{R}^{\mathcal{K} \times D}$. $\text{PE}(\cdot)$ denotes the sinusoidal position encoding. Notably, each intention query is responsible for predicting trajectories for a specific motion mode,

[1]To streamline the illustration of the motion decoder, we simplify the two components of the motion query pair in our previous version [46] by using the new concept of intention query.

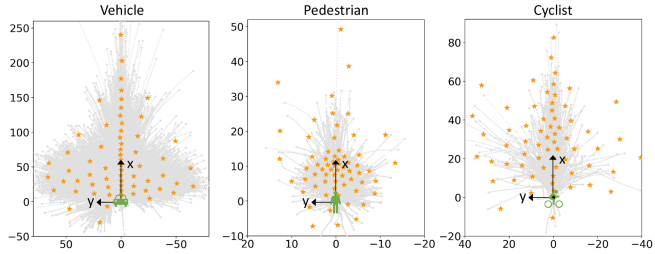


Fig. 3. Distribution of intention points for each category, where the intention points are shown as orange stars. The gray dotted lines indicate the distribution of ground-truth trajectories for each category, and note that only 10% ground-truth trajectories in the training dataset are drawn in the figure for better visualization.

which stabilizes the training process and facilitates multimodal trajectory prediction since each motion mode possesses its own learnable embedding. Owing to their learnable and adaptive properties, we require only a minimal number of queries (e.g., 64 queries in our setting) for efficient intention localization, rather than employing densely-placed goal candidates [21], [67] to cover the agents' destinations.

Scene Context Aggregation With Intention Query: These intention queries are considered as the learnable query embedding of the transformer decoder layer for aggregating context features from the encoded agent features and map features. Specifically, in each transformer decoder layer, we first apply the self-attention module to propagate information among $\mathcal{K}$ intention queries as follows:

$$\begin{aligned} F'^{(\text{s})}_{\text{I}}[i] = \text{MHSA}\big(&\text{Q: } F_{\text{I}}^{(\text{s})}[i] + E_{\text{I}}^{(\text{s})}[i], \\ &\text{K: } \left\{F_{\text{I}}^{(\text{s})}[j] + E_{\text{I}}^{(\text{s})}[j]\right\}_{j=1}^{\mathcal{K}}, \\ &\text{V: } \left\{F_{\text{I}}^{(\text{s})}[j]\right\}_{j=1}^{\mathcal{K}}\big), \end{aligned} \tag{5}$$

where $i \in \{1, \ldots, \mathcal{K}\}$. $F_{\text{I}}^{(\text{s})} \in \mathbb{R}^{\mathcal{K} \times D}$ is the query content feature from the previous transformer decoder layer, and it is initialized as zero in the first transformer decoder layer. $F'^{(\text{s})}_{\text{I}} \in \mathbb{R}^{\mathcal{K} \times D}$ indicates the updated query content feature. Next, to aggregate scene context features from the encoder network, inspired by [32], [36], we concatenate content features and position embedding for both query and key to decouple their contributions to the attention weights. Thus, the cross-attention module can be formulated as follows:

$$\begin{aligned} F''^{(\text{s})}_{\text{I}}[i] = \text{MHCA}\big(&\text{Q: } [F'^{(\text{s})}_{\text{I}}[i], E_{\text{I}}^{(\text{s})}[i]], \\ &\text{K: } [F_{\text{A}}^{(\text{g})}, \text{PE}(P_{\text{A}}^{(\text{g})})] \cup [F_{\text{M}}^{(\text{g})}, \text{PE}(P_{\text{M}}^{(\text{g})})], \\ &\text{V: } F_{\text{A}}^{(\text{g})} \cup F_{\text{M}}^{(\text{g})}\big), \end{aligned} \tag{6}$$

where $i \in \{1, \ldots, \mathcal{K}\}$. $\text{MHCA}(\cdot_{\text{query}}, \cdot_{\text{key}}, \cdot_{\text{value}})$ denotes the multi-head cross-attention layer [51]. The sign "$[\cdot, \cdot]$" indicates feature concatenation, and "$\cup$" combines the agent tokens and map tokens as the key and value of the cross-attention module. Finally, $F''^{(\text{s})}_{\text{I}} \in \mathbb{R}^{\mathcal{K} \times D}$ is the final updated query content feature in this transformer decoder layer.

Additionally, for each intention query, we introduce the dynamic map collection strategy to extract fine-grained trajectory

features by querying map features from a trajectory-aligned local region. Specifically, by adopting such a module, the key and value of the map tokens in (6) are restricted to a local region by gathering the polylines whose centers are nearest to the predicted trajectory of the current intention query. As the agent's behavior is largely influenced by road maps, this local movement refinement strategy enables a continuous focus on the most recent local context information for iterative motion refinement.

Global Intention Localization: By considering different motion modes with different learnable queries, the intention queries capture representative features $F''^{(s)}_{I} \in \mathbb{R}^{\mathcal{K}\times D}$ to model the focal agent's future motion. Thus, we propose to coarsely localize the agent's intention by predicting the occurrence probability of each intention point as follows:

$$p = \text{MLP}\left(F''^{(s)}_{I}\right), \tag{7}$$

where $p \in \mathbb{R}^{\mathcal{K}}$ is a probability distribution to model the potential future intention of the focal agent.

Local Movement Refinement: To complement the coarse global intention localization, we further predict the detailed future trajectory for each intention query as follows:

$$Z = \text{MLP}\left(F''^{(s)}_{I}\right), \tag{8}$$

where $Z \in \mathbb{R}^{\mathcal{K}\times(T\times 5)}$ indicates the $\mathcal{K}$ predicted future trajectories, and each of them has T future frames. "5" indicates that we model the uncertainty of each trajectory waypoint with Gaussian distribution as $\mathcal{N}(\mu_x, \sigma_x; \mu_y, \sigma_y; \rho)$.

As the query content feature $F''^{(s)}_{I}$ will be constantly propagated to the next transformer decoder layer as the new query content feature, the predicted future trajectories can be iteratively refined with multiple stacked transformer decoder layers by continually aggregating scene context features from the encoder network.

C. Multimodal Prediction With Gaussian Mixture Model

As the behaviors of the agents are highly multimodal, we follow [10], [50] to represent the distribution of predicted trajectories with Gaussian Mixture Model (GMM) at each time step. Specifically, for a specific future time step i, MTR will predict $\mathcal{K}$ candidate goal positions with distribution $\mathcal{N}_{1:\mathcal{K}}(\mu_x, \sigma_x; \mu_y, \sigma_y; \rho)$ and probability distribution $p \in \mathbb{R}^{\mathcal{K}}$. The predicted distribution of the focal agent's position at time step i can be formulated as a GMM with $\mathcal{K}$ components

$$\mathcal{P}_i(o) = \sum_{k=1}^{\mathcal{K}} p_k \cdot f_k\left(o_x - \mu_x, o_y - \mu_y\right), \tag{9}$$

where $f_k(\cdot,\cdot)$ is the probability density function of the kth component of this GMM, and $\mathcal{P}_i(o)$ is the occurrence probability density of the agent at spatial position $o \in \mathbb{R}^2$. The predicted trajectories can be generated by simply extracting the predicted centers of Gaussian components.

Training Loss: Given the predicted Gaussian Mixture Models for a specific future time step, we adopt negative log-likelihood loss to maximize the likelihood of the agent's ground-truth position $(\hat{Y}_x, \hat{Y}_y)$ at this time step, and the detailed loss can be formulated as

$$L_{\text{GMM}} = -\log f_h\left(\hat{Y}_x - \mu_x, \hat{Y}_y - \mu_y\right) - \log(p_h), \tag{10}$$

where $f_h(\hat{Y}_x - \mu_x, \hat{Y}_y - \mu_y)$ is the selected positive Gaussian component for optimization. Here the positive Gaussian component is selected by finding the closest intention query with the endpoint of this GT trajectory. p_h is the predicted probability of this selected positive Gaussian component, and we adopt cross entropy loss in the above equation to maximize the probability of the selected positive Gaussian component. The final loss of our framework is denoted as

$$L_{\text{SUM}} = L_{\text{GMM}} + L_{\text{DMP}}, \tag{11}$$

where L_{DMP} is the $L1$ regression loss on the outputs of (3).

IV. MTR++: Multi-Agent Motion Prediction

The above MTR framework proposed for multimodal motion prediction has demonstrated state-of-the-art performance. However, its scene context modeling module adopts the focal-agent-centric strategy commonly found in previous works [21], [50], [67], which encodes the scene context separately for each focal agent, leading to computational inefficiencies when predicting motion for multiple agents. Although the Scene Transformer model [39] presents a shared context encoding strategy for predicting trajectories of multiple agents, it still centers the scene around a specific agent, potentially limiting its performance for off-center agents due to uneven distribution of shared context information.

To address the aforementioned challenges, we introduce an enhanced version of the MTR framework, denoted as MTR++. As shown in Fig. 4, the MTR++ framework enables simultaneous motion prediction of multiple agents via shared symmetric scene context modeling and mutually-guided intention querying. We elaborate these two improvements in Sections IV-A and IV-B, respectively.

A. Symmetric Scene Context Modeling for All Agents

To improve the efficiency of predicting future trajectories of multiple agents simultaneously, we propose a symmetric scene context modeling module that employs a shared context encoder to encode complex multimodal scene context information for all agents. In contrast to most existing methods that center the scene around a particular agent, our approach encodes the entire scene symmetrically for each agent. As a result, the encoded scene context features can be directly utilized for predicting the motion of any agent by attaching a motion decoder network.

Input Representation With Polyline-Centric Encoding: We employ the same vectorized representation as in Section III-A to encode the input context features. However, instead of normalizing all inputs to the global coordinate system centered on one focal agent, we encode the feature of each polyline in a polyline-centric local coordinate system (see Fig. 5). Specifically, we modify the polyline feature encoding process in (1) by

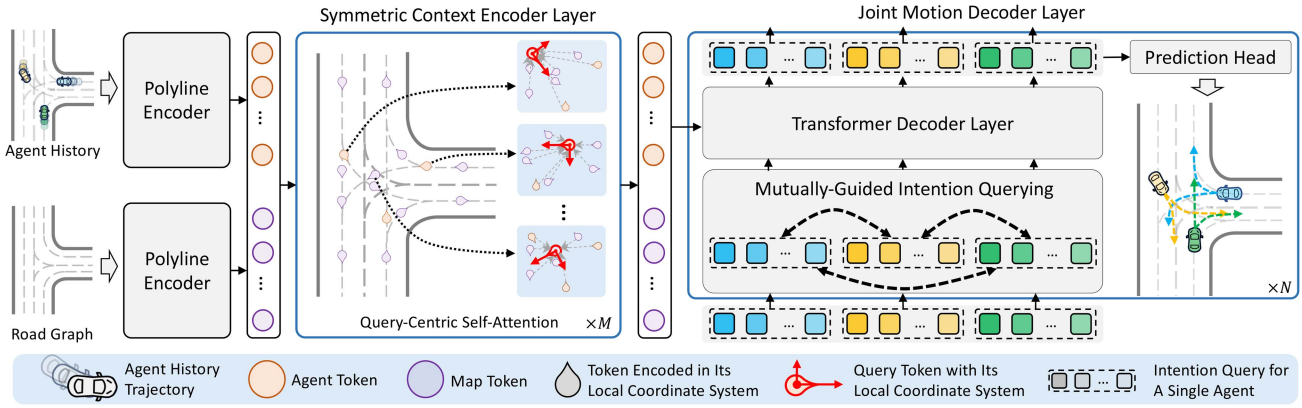


Fig. 4. Architecture of the MTR++ framework builds upon the initial MTR framework and introduces several enhancements. In the MTR++ framework, we introduce the symmetric context encoder layer, which facilitates the understanding of relationships among tokens within their respective local coordinate systems. By incorporating these symmetrically encoded token features as input, the MTR++ framework employs a joint motion decoder that leverages multiple sets of intention queries. This enables the simultaneous prediction of future trajectories for multiple agents, with the mutually-guided intention querying module facilitating the interaction of future behaviors among different agents. As a result, the MTR++ framework generates more scene-compliant future trajectories, enhancing the overall predictive capabilities of the MTR framework.

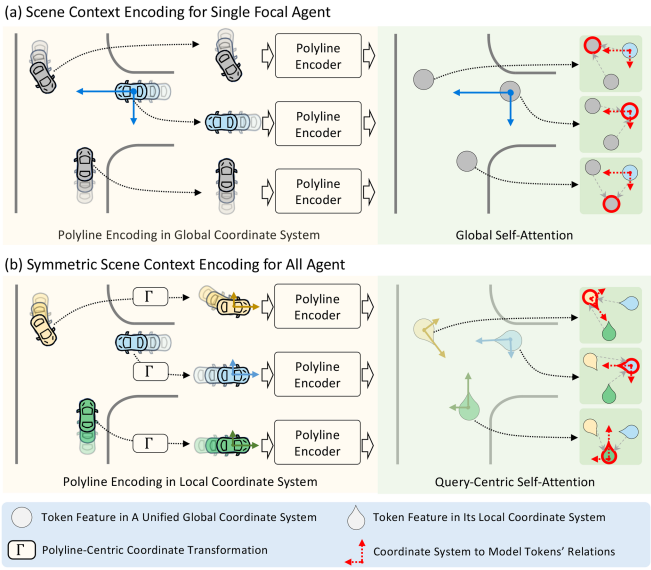


Fig. 5. Comparison of two different scene context encoding modules in the MTR and MTR++ frameworks. The MTR framework adopts the scene context encoding for a single focal agent, where both the polyline-wise features and tokens' relationship are encoded in a global coordinate system. In contrast, the MTR++ framework encodes both the polyline-wise features and their relationship in their respective local coordinate system via the novel query-centric self-attention module, thus enabling simultaneous motion prediction of multiple agents.

incorporating the coordinate transformation function, denoted as $\Gamma(\cdot)$, as follows:

$$F_{\mathrm{A}}^{(l)} = \phi\left(\mathrm{MLP}\left(\Gamma(S_{\mathrm{A}}^{(\mathrm{g})})\right)\right), \qquad F_{\mathrm{M}}^{(l)} = \phi\left(\mathrm{MLP}\left(\Gamma(S_{\mathrm{M}}^{(\mathrm{g})})\right)\right), \tag{12}$$

where $\Gamma(\cdot)$ transforms the polyline features from an arbitrary global coordinate system to the polyline-centric local coordinate system. Concretely, we use the latest position and moving direction of each agent to determine the local coordinate system of their corresponding polyline, while for the map polylines, we calculate the geometry center and tangent direction of each polyline to determine their local coordinate system.

The encoded features $F_{\mathrm{A}}^{(l)} \in \mathbb{R}^{N_a \times D}$ and $F_{\mathrm{M}}^{(l)} \in \mathbb{R}^{N_m \times D}$ (where "l" indicating the local reference frame) capture the polyline-wise features for the agent history states and map polylines, respectively. Importantly, these polyline features are encoded in their own local coordinate system, independent of any global coordinate system. This provides input token features that are decoupled from the global coordinate system and enables the symmetric modeling of token relations in the subsequent step.

Attribute Definition of Polyline Tokens: The features $F_{\mathrm{A}}^{(l)}$ and $F_{\mathrm{M}}^{(l)}$ are considered as input tokens in the subsequent transformer network, and their features are concatenated to form the input token feature matrix $F_{\mathrm{AM}}^{(l)} = [F_{\mathrm{A}}^{(l)}, F_{\mathrm{M}}^{(l)}] \in \mathbb{R}^{(N_a+N_m)\times D}$. As in Section III-A, the global positions of these tokens are denoted as $P_{\mathrm{AM}}^{(\mathrm{g})} \in \mathbb{R}^{(N_a+N_m)\times 2}$, which can be defined in an arbitrary global coordinate system. Additionally, each token is associated with a heading direction attribute $H_{\mathrm{AM}}^{(\mathrm{g})} \in \mathbb{R}^{(N_a+N_m)\times 1}$, which is defined similarly to the direction definition as in the transformation function $\Gamma(\cdot)$ presented in (12).

Symmetric Scene Context Modeling With Query-Centric Self-Attention: In our previous MTR framework, we model the relationship between the input token features using a self-attention module (2) that depends on a global coordinate system centered on a single focal agent. However, this approach hindered the performance of motion prediction for other agents. To address this limitation, we propose a *query-centric self-attention* module, which models the relationship between all tokens in a symmetric manner, decoupled from any global coordinate system.

Specifically, to explore the relationship between a query token and other tokens in its specific local coordinate system, we perform the attention mechanism separately for each query token. For instance, let us consider the ith token as the query.

We convert the coordinates and directions of all tokens into the local coordinate system of the query token

$$R_{\text{AM}}^{(\text{pos})}[i,j] = (P_{\text{AM}}^{(\text{g})}[j] - P_{\text{AM}}^{(\text{g})}[i]) \begin{bmatrix} \cos H_{\text{AM}}^{(\text{g})}[i] & -\sin H_{\text{AM}}^{(\text{g})}[i] \\ \sin H_{\text{AM}}^{(\text{g})}[i] & \cos H_{\text{AM}}^{(\text{g})}[i] \end{bmatrix},$$

$$R_{\text{AM}}^{(\text{ang})}[i,j] = H_{\text{AM}}^{(\text{g})}[j] - H_{\text{AM}}^{(\text{g})}[i], \tag{13}$$

where $i \in \{1, \ldots, N_a + N_m\}$, and $j \in \Omega(i)$ indicating the index of its neighboring tokens. $R_{\text{AM}}^{(\text{pos})}[i,j] \in \mathbb{R}^2$ and $R_{\text{AM}}^{(\text{ang})}[i,j] \in \mathbb{R}$ indicate the jth token's relative position and direction in the local coordinate system of the ith query token. We then perform the query-centric self-attention mechanism as follows:

$$\begin{aligned} F'^{(l)}_{\text{AM}}[i] = \text{MHSA}\big(&\text{Q: } [F_{\text{AM}}^{(l)}[i], \text{PE}(R_{\text{AM}}[i,i])], \\ &\text{K: } \big\{[F_{\text{AM}}^{(l)}[j], \text{PE}(R_{\text{AM}}[i,j])]\big\}_{j\in\Omega(i)}, \\ &\text{V: } \big\{F_{\text{AM}}^{(l)}[j] + \text{PE}(R_{\text{AM}}[i,j])\big\}_{j\in\Omega(i)}\big), \end{aligned} \tag{14}$$

where $\text{PE}(R_{\text{AM}}[i,j])$ indicates the sinusoidal positional encoding of both $R_{\text{AM}}^{(\text{pos})}[i,j]$ and $R_{\text{AM}}^{(\text{ang})}[i,j]$. This query-centric self-attention mechanism models the token relationship in a symmetric manner by integrating the global-coordinate-decoupled token feature $F_{\text{AM}}^{(l)}$ and relative coordinate R_{AM} based on the query token.

Note that the computational cost of the proposed query-centric self-attention module in (14) is comparable to that of the self-attention module in (2). The key advantage of the proposed module is that it enables the symmetric encoding of scene context features for each input token, such as each agent, thus allowing the encoded features to be used for predicting the motion of any input agent. This feature enables a shared scene context encoder for simultaneous prediction of the motion of multiple agents.

B. Joint Motion Decoder With Mutually-Guided Queries

The MTR++ framework utilizes symmetrically encoded scene context features, which are fed to the motion decoder as described in Section III-B, to enable simultaneous motion prediction for multiple focal agents. This simultaneous motion prediction allows for the exploration of future behavior interactions among the agents, which is crucial for making more accurate and scene-compliant motion predictions.

Mutually-Guided Intention Querying of Multiple Agents: To enhance the accuracy of motion prediction by enabling agents to interact and influence each other's behavior, as shown in Fig. 6, we propose a *mutually-guided intention querying* module. However, building such interaction is non-trivial since the intention queries of different focal agents are encoded in their own local coordinate system as in (4). To maintain the local-encoded features of intention queries while also establishing the spatial relationship among them, we adopt the previously introduced query-centric self-attention module, similar to the one used in Section IV-A, to enable the information interaction among all intention queries.

Specifically, to predict future trajectories for N_o focal agents, the motion decoding process is conducted simultaneously in the local coordinate system centered at each focal agent.

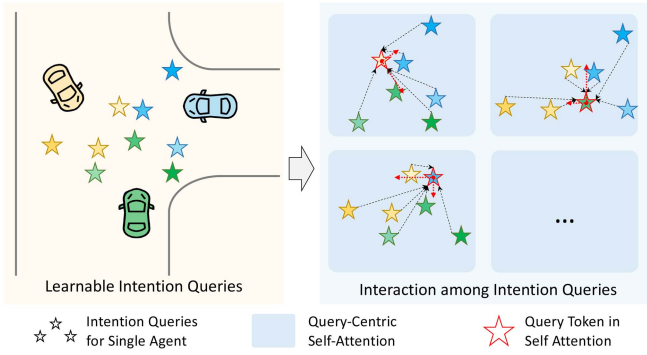


Fig. 6. Illustration of the mutually-guided intention querying module.

The intention queries for the focal agents are represented as $E_{\text{I}}^{(\text{m})} \in \mathbb{R}^{N_o \times \mathcal{K} \times D}$ (where "m" indicating multiple focal agents), wherein the intention queries for different focal agents are encoded using (4) with the same intention points $I^{(\text{s})} \in \mathbb{R}^{\mathcal{K}\times 2}$.

However, as the intention points are defined in their respective local coordinate systems, in order to facilitate information propagation among the intention queries of different focal agents, we first transform their intention points into the same global coordinate system based on the global positions $P_{\text{O}}^{(\text{g})} \in \mathbb{R}^{N_o \times 2}$ and moving directions $H_{\text{O}}^{(\text{g})} \in \mathbb{R}^{N_o \times 1}$ of the focal agents, as follows:

$$P_{\text{I}}^{(\text{m})}[t] = I^{(\text{s})} \begin{bmatrix} \cos H_{\text{O}}^{(\text{g})}[t] & \sin H_{\text{O}}^{(\text{g})}[t] \\ -\sin H_{\text{O}}^{(\text{g})}[t] & \cos H_{\text{O}}^{(\text{g})}[t] \end{bmatrix} + P_{\text{O}}^{(\text{g})}[t], \tag{15}$$

where $t \in \{1, \ldots, N_o\}$ and $P_{\text{I}}^{(\text{m})} \in \mathbb{R}^{N_o \times \mathcal{K} \times 2}$. To build the information interaction among all intention queries of multiple agents, we re-organize the intention points and intention queries as $P_{\text{I}}^{(\text{m})} \in \mathbb{R}^{(N_o\mathcal{K})\times 2}$ and $E_{\text{I}}^{(\text{m})} \in \mathbb{R}^{(N_o\mathcal{K})\times D}$, respectively. Meanwhile, we also assign the heading direction $H_{\text{I}}^{(\text{m})} \in \mathbb{R}^{(N_o\mathcal{K})\times 1}$ for the intention queries for calculating their relative spatial relationship, where the $\mathcal{K}$ intention queries of the tth focal agent share the same heading direction as its moving direction $H_{\text{O}}^{(\text{g})}[t]$.

Thus, following (13), when considering the ith intention query as the query token, we transform the coordinates and directions of all intention queries to the local coordinate system of the ith query token, as follows:

$$R_{\text{I}}^{(\text{pos})}[i,j] = (P_{\text{I}}^{(\text{m})}[j] - P_{\text{I}}^{(\text{m})}[i]) \begin{bmatrix} \cos H_{\text{I}}^{(\text{m})}[i] & -\sin H_{\text{I}}^{(\text{m})}[i] \\ \sin H_{\text{I}}^{(\text{m})}[i] & \cos H_{\text{I}}^{(\text{m})}[i] \end{bmatrix},$$

$$R_{\text{I}}^{(\text{ang})}[i,j] = H_{\text{I}}^{(\text{m})}[j] - H_{\text{I}}^{(\text{m})}[i], \tag{16}$$

where $i \in \{1, \ldots, N_o\mathcal{K}\}$, and $j \in \Omega(i)$ indicating the index of its neighboring tokens. Then, we apply the query-centric self-attention module on all intention queries as follows:

$$\begin{aligned} F'^{(\text{m})}_{\text{I}}[i] = \text{MHSA}\big(&\text{Q: } [F_{\text{I}}^{(\text{m})}[i] + E_{\text{I}}^{(\text{m})}[i], \text{PE}(R_{\text{I}}[i,i])], \\ &\text{K: } \{[F_{\text{I}}^{(\text{m})}[j] + E_{\text{I}}^{(\text{m})}[j], \text{PE}(R_{\text{I}}[i,j])]\}_{j\in\Omega(i)}, \end{aligned}$$

$$\text{V}: \{F_{\text{I}}^{(\text{m})}[j] + E_{\text{I}}^{(\text{m})}[j] + \text{PE}(R_{\text{I}}[i,j])\}_{j \in \Omega(i)}\big), \tag{17}$$

where $i \in \{1, \ldots, N_o\mathcal{K}\}$, and $F_{\text{I}}^{(\text{m})} \in \mathbb{R}^{(N_o\mathcal{K}) \times D}$ indicates the query content feature from the previous transformer decoder layer and is initialized as zero in the first decoder layer.

Finally, the updated query content feature $F'^{(\text{m})}_{\text{I}} \in \mathbb{R}^{N_o \times \mathcal{K} \times D}$ will be utilized individually for the subsequent scene context aggregation of each focal agent. This aggregation process is the same as described in (5) and (6) in the MTR framework. It is worth noting that the positional encoding for the encoded scene elements from the context encoder is defined in the local coordinate system of each focal agent. These resulting query features are then fed into the prediction head, which generates future trajectories for each focal agent. By establishing this information propagation process, the intention queries of multiple agents are guided by each other during the multimodal motion decoding process, ultimately resulting in more informed and realistic predictions of their future trajectories.

V. Experiments

A. Experimental Setup

Dataset and Metrics: We mainly evaluate our approach using the Waymo Open Motion Dataset (WOMD) [15], a large-scale dataset that captures diverse traffic scenes with interesting interactions among agents. There are two tasks in WOMD with separate evaluation metrics: (1) The *marginal motion prediction challenge* that independently evaluates the predicted motion of each agent (up to 8 agents per scene). (2) The *joint motion prediction challenge* that needs to predict the joint future positions of 2 interacting agents for evaluation. For both tasks, the dataset provides 1 s of history data and aims to predict 6 marginal or joint trajectories of the agents for 8 seconds into the future. The dataset contains $487k$ training scenes, and approximately $44k$ validation scenes and $44k$ testing scenes for each challenge. We utilize the official evaluation tool, which calculates important metrics such as mAP and miss rate, as used in the official WOMD leaderboards [56], [58].

In addition to the WOMD, we also evaluate our approach on the Argoverse 2 Motion Forecasting Dataset [59], another large-scale motion prediction dataset. It contains 250,000 scenarios for training and validation. The model needs to take the history five seconds of each scenario as input and predict the six-second future trajectories of one interested agent, where HDMap is always available to provide map context information. We also utilize the official evaluation tool to calculate the miss rate as the main metric.

Implementation Details: For both the MTR and MTR++ frameworks, we stack 6 transformer encoder layers for context encoding. The road map is represented as multiple polylines, where each polyline contains up to 20 map points (about $10m$ in WOMD). We select $N_m = 768$ nearest map polylines around the interested agents. The number of neighbors in the encoder's local self-attention is set to 16. The hidden feature dimension is set as $D = 256$. For the decoder modules, we stack 6 decoder layers. For dynamic map collection, we collect the closest 128 map polylines from the context encoder for iterative motion refinement. By default, we utilize 64 motion query pairs where their intention points are generated by conducting the k-means clustering algorithm on the training dataset. The number of neighbors for the query-centric self-attention module is set to 16 for the MTR++ framework. To generate 6 future trajectories for evaluation, we use non-maximum suppression (NMS) to select the top 6 predictions from 64 predicted trajectories by calculating the distances between their endpoints, and the distance threshold is set as $2.5m$. More implementation details of the initial MTR framework can be found in our open-source codebase: https://github.com/sshaoshuai/MTR.

Training Details: Our model is trained in an end-to-end manner by AdamW optimizer with a learning rate of 0.0001, a weight decay of 0.01, and a batch size of 80 scenes. We train the model for 30 epochs with 8 GPUs, and the learning rate is decayed by a factor of 0.5 every 2 epochs from epoch 20.

B. Main Results

Performance Comparison for Marginal Motion Prediction: We evaluate the marginal motion prediction performance of our MTR frameworks by comparing them with leading-edge research on the WOMD test set. As presented in Table I, our initial MTR framework already surpasses previous state-of-the-art approaches [21], [27], [39] with significant improvements. It achieves an mAP increase of $+8.48\%$ and reduces the miss rate from 15.11% to 13.51%. Furthermore, our latest MTR++ framework further enhances the performance compared to MTR on all metrics. Particularly, it achieves a $+2.00\%$ improvement in mAP, showcasing its ability to generate more confident multimodal future trajectories by jointly considering the future behaviors of multiple agents.

Additionally, we also adopt a simple model ensemble strategy, combining predictions from multiple models and employing non-maximum-suppression (NMS) to remove redundant predictions. By adopting this ensemble strategy to diverse variants of our MTR frameworks (e.g., more decoder layers, different number of queries, larger hidden dimension), our approach significantly outperforms the previous state-of-the-art ensemble result [50], increasing the mAP by $+5.42\%$ and reducing the miss rate from 13.40% to 11.22%.

Notably, our MTR and MTR++ frameworks have secured the first-place positions in the highly-competitive Waymo Motion Prediction Challenge in 2022 [57] and 2023 [58], respectively. As of May 30, 2023, our MTR++ framework holds the 1st rank on the motion prediction leaderboard of WOMD [58], outperforming other works by a significant margin. These notable achievements highlight the effectiveness of the MTR frameworks.

Performance Comparison for Joint Motion Prediction: We also evaluate the proposed MTR frameworks on the joint motion prediction benchmark, merging the marginal predictions of two interacting agents into a joint prediction as explained in [7], [15], [47]. We select the top 6 joint predictions from 36 potential combinations of these agents, with the confidence of

TABLE I
PERFORMANCE COMPARISON OF MARGINAL MOTION PREDICTION ON THE TEST AND VALIDATION SET OF WAYMO OPEN MOTION DATASET

	Method	Reference	minADE ↓	minFDE ↓	Miss Rate ↓	**mAP ↑**
Test	MotionCNN [28]	CVPRw 2021	0.7400	1.4936	0.2091	0.2136
	ReCoAt [68]	CVPRw 2021	0.7703	1.6668	0.2437	0.2711
	DenseTNT [21]	ICCV 2021	1.0387	1.5514	0.1573	0.3281
	SceneTransformer [39]	ICLR 2022	0.6117	1.2116	0.1564	0.2788
	HDGT [27]	Arxiv 2022	0.5933	1.2055	0.1511	0.2854
	MTR (Ours)	NeurIPS 2022	0.6050	1.2207	0.1351	0.4129
	MTR++ (Ours)	-	**0.5906**	**1.1939**	**0.1298**	**0.4329**
	†MultiPath++ [50]	ICRA 2022	*0.5557*	*1.1577*	*0.1340*	*0.4092*
	†MTR++_Ens (Ours)	-	*0.5581*	*1.1166*	*0.1122*	*0.4634*
Val	MTR (Ours)	NeurIPS 2022	0.6046	1.2251	0.1366	0.4164
	MTR++ (Ours)	-	**0.5912**	**1.1986**	**0.1296**	**0.4351**

†: The results are shown in italic for reference since their performance is achieved with model ensemble techniques.

TABLE II
PERFORMANCE COMPARISON OF JOINT MOTION PREDICTION ON THE INTERACTIVE VALIDATION AND TEST SET OF WAYMO OPEN MOTION DATASET

	Method	Reference	minADE ↓	minFDE ↓	Miss Rate ↓	**mAP ↑**
Test	Waymo LSTM baseline [15]	ICCV 2021	1.9056	5.0278	0.7750	0.0524
	HeatIRm4 [38]	CVPRw 2021	1.4197	3.2595	0.7224	0.0844
	AIR^2 [60]	CVPRw 2021	1.3165	2.7138	0.6230	0.0963
	SceneTransformer [39]	ICLR 2022	0.9774	2.1892	0.4942	0.1192
	M2I [47]	CVPR 2022	1.3506	2.8325	0.5538	0.1239
	MTR (Ours)	NeurIPS 2022	0.9181	2.0633	0.4411	0.2037
	MTR++ (Ours)	-	**0.8795**	**1.9509**	**0.4143**	**0.2326**
Val	MTR (Ours)	NeurIPS 2022	0.9132	2.0536	0.4372	0.1992
	MTR++ (Ours)	-	**0.8859**	**1.9712**	**0.4106**	**0.2398**

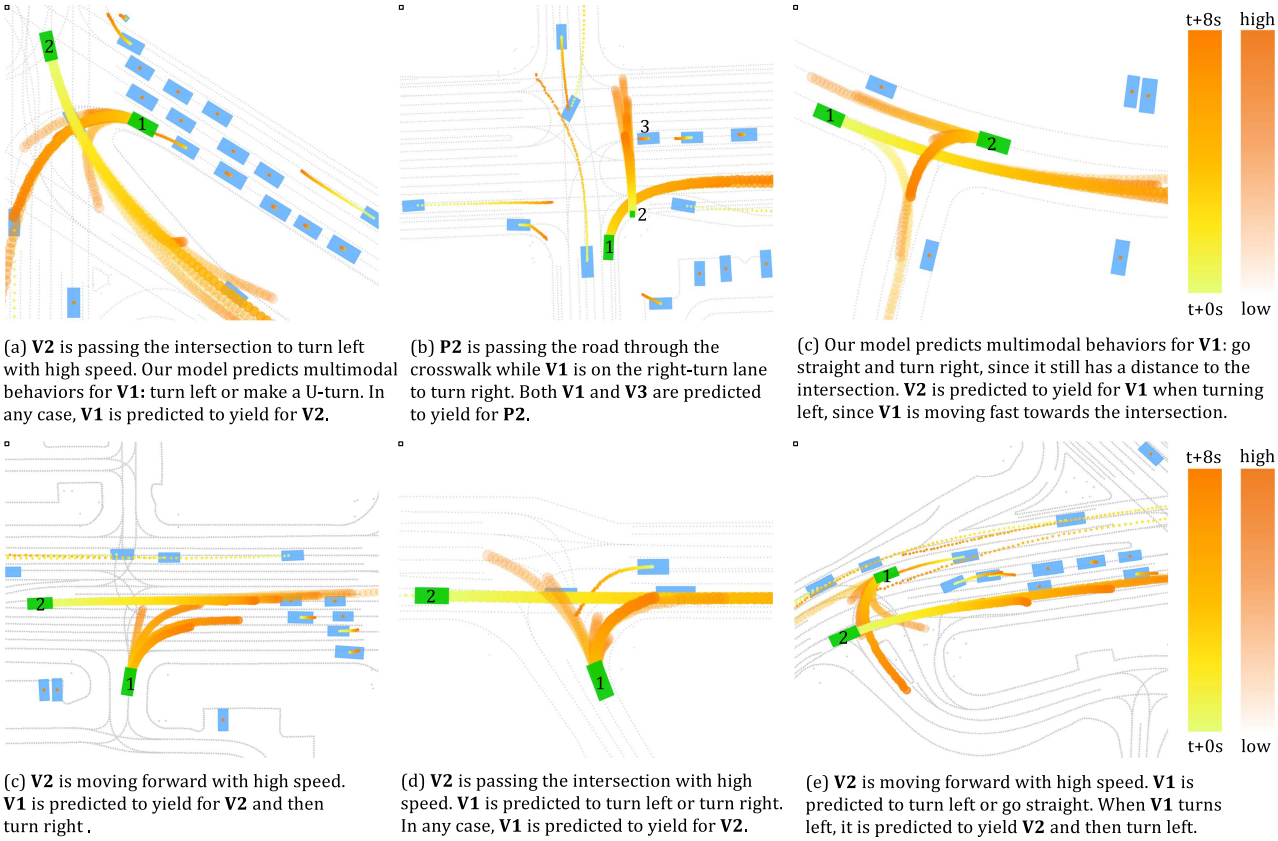
each combination being the product of marginal probabilities. As indicated in Table II, our initial MTR framework already surpasses state-of-the-art approaches [39], [47] by substantial margins on all measures, reducing the miss rate from 49.42% to 44.11% and enhancing the mAP from 12.39% to 20.37%. Furthermore, our advanced MTR++ framework, which allows us to concurrently predict future motion for two interactive agents with shared context encoding, amplifies the robust performance of MTR across all metrics, achieving an increase of $+2.89\%$ in terms of mAP and reducing the miss rate by 2.68%. The extraordinary performance enhancements of the MTR++ framework emphasize that, through the adoption of symmetric scene context encoding and mutually-guided intention querying, our MTR++ framework can accurately predict future trajectories that exhibit scene consistency among highly interacting agents. Additionally, we also provide some qualitative results in Fig. 7 to show our predictions in complicated interacting scenarios. Notably, as of May 30, 2023, our MTR++ framework holds the 1st rank on the joint motion prediction leaderboard of WOMD [56].

Moreover, we provide the performance comparison in terms of the number of parameters and inference latency in Table III. To enable a comparison of performance and latency with a similar parameter count, we have provided a streamlined version of MTR++ (notated as "MTR++ (light)") by diminishing the number of encoder layers to 2 and decoder layers to 3. As indicated in Table III, our lightweight MTR++ surpasses existing approaches markedly, enhancing the mAP from 32.81% to 38.96%. When compared with SceneTransformer [39], our MTR++ (light) not only employs fewer parameters (15.3 M down to 11.7 M) but also attains superior performance (27.88% boosting to 38.96%). When compared with DenseTNT [21], albeit our MTR++ (light) utilizes more parameters, it operates considerably faster (67 ms versus 540 ms). This enhanced speed can be attributed to our method sidestepping the densely sampled goal points present in DenseTNT. When compared with HDGT [27], our MTR++ (light) runs significantly faster (67 ms as opposed to 1320 ms). This efficiency arises from our usage of the well-optimized native Transformer module, whereas HDGT deploys a custom graph neural network. Furthermore, our standard MTR and MTR++ elevate performance through the use of additional parameters, while simultaneously maintaining inference latencies significantly lower than the existing open-source approaches [21], [27].

TABLE III
PERFORMANCE IS COMPARED IN TERMS OF THE NUMBER OF PARAMETERS AND INFERENCE LATENCY FOR MULTI-AGENT PREDICTION, UTILIZING THE WAYMO OPEN MOTION DATASET

Method	Number of Parameters	Inference Latency	Miss Rate ↓	**mAP ↑**
†SceneTransformer [39]	15.3M	52ms (V100)	0.1564	0.2788
DenseTNT [21]	1.1M	540ms	0.1573	0.3281
HDGT [27]	12.1M	1320ms	0.1511	0.2854
MTR++ (light)	11.7M	67ms	0.1430	0.3896
MTR	65.8M	193ms	0.1351	0.4129
MTR++	86.6M	118ms	**0.1298**	**0.4329**

Latency is estimated by mandating each model to predict the motion of 32 agents per scenario, leveraging an NVIDIA Quadro RTX 8000 GPU. †: the number of parameters and latency are reported in the respective source paper, using NVIDIA V100 GPU.



(a) **V2** is passing the intersection to turn left with high speed. Our model predicts multimodal behaviors for **V1:** turn left or make a U-turn. In any case, **V1** is predicted to yield for **V2.**

(b) **P2** is passing the road through the crosswalk while **V1** is on the right-turn lane to turn right. Both **V1** and **V3** are predicted to yield for **P2.**

(c) Our model predicts multimodal behaviors for **V1**: go straight and turn right, since it still has a distance to the intersection. **V2** is predicted to yield for **V1** when turning left, since **V1** is moving fast towards the intersection.

(c) **V2** is moving forward with high speed. **V1** is predicted to yield for **V2** and then turn right .

(d) **V2** is passing the intersection with high speed. **V1** is predicted to turn left or turn right. In any case, **V1** is predicted to yield for **V2.**

(e) **V2** is moving forward with high speed. **V1** is predicted to turn left or go straight. When **V1** turns left, it is predicted to yield **V2** and then turn left.

Fig. 7. Qualitative results of MTR frameworks on WOMD. There are two interested agents in each scene (green rectangle), where our model predicts 6 multimodal future trajectories for each of them. For other agents (blue rectangle), a single trajectory is predicted by the dense future prediction module. We use gradient color to visualize the trajectory waypoints at different future time steps, and trajectory confidence is visualized by setting different transparent. Abbreviation: Vehicle (V), Pedestrian (P).

TABLE IV
PERFORMANCE COMPARISON ON THE TEST SET LEADERBOARD OF THE ARGOVERSE 2 DATASET

Method	Miss Rate ↓ (K=6)	Miss Rate ↓ (K=1)	brier-minFDE ↓ (K=6)
MTR++ (Ours)	**0.14**	**0.56**	**1.88**
MTR (Ours)	0.15	0.58	1.98
TENET [55]	0.19	0.61	1.90
OPPred	0.19	0.60	1.92
Qml	0.19	0.62	1.95
GANet	0.17	0.60	1.97
VI LaneIter	0.19	0.61	2.00
QCNet	0.21	0.60	2.14
THOMAS [20]	0.20	0.64	2.16
HDGT [27]	0.21	0.66	2.24
GNA	0.29	0.71	2.45
vilab	0.29	0.71	2.47

K is the number of predicted trajectories for calculating the evaluation metrics.

Performance Comparison on the Argoverse 2 Dataset: As shown in Table IV, we also provide the performance comparison of our approach on the Argoverse 2 dataset for reference. We compare our approach with the top-10 submissions on the leaderboard of Argoverse 2 dataset [1] at the time of our MTR framework submission. These submissions, primarily developed for the Argoverse 2 Motion Forecasting Competition 2022, represent highly competitive approaches. Notably, our MTR framework achieves new state-of-the-art performance with remarkable improvements in miss-rate-related metrics. Moreover, our MTR++ further surpasses the performance of both MTR and other existing approaches across all three metrics, thereby highlighting the exceptional generalizability and robustness of our approach.

C. Ablation Study

We study the effectiveness of each component in our MTR/MTR++ frameworks. For efficiently conducting ablation experiments, we uniformly sampled 20% frames (about 97 k scenes) from the WOMD training set according to their default order,[2] and we empirically find that it has similar distribution with the full training set. All models are evaluated with marginal motion prediction metric on the validation set of WOMD.

Effects of the Learnable Intention Query: We investigate the effectiveness of different strategies for generating future trajectories based on encoded context features. These strategies include the simple MLP head [27], [39], the goal-based head [21], the head with 6 latent anchor embeddings [50], and the head with the learnable intention query. The first four rows of Table V illustrate the performance comparison of these

[2]The detailed training data split can be found in our open-source codebase: https://github.com/sshaoshuai/MTR

TABLE V
EFFECTS OF DIFFERENT STRATEGIES FOR GENERATING TRAJECTORIES FROM ENCODED CONTEXT FEATURES IN THE MTR FRAMEWORK

Trajectory Generation	Iterative Refinement	minADE ↓	Miss Rate ↓	**mAP ↑**
MLP		0.6870	0.2103	0.2747
Dense Goals		1.0544	0.1936	0.2912
Latent Embedding		0.6564	0.1882	0.2826
Intention Query		0.6885	0.1723	0.3379
Intention Query	✓	**0.6557**	**0.1575**	**0.3539**

TABLE VI
EFFECTS OF DIFFERENT STRATEGIES FOR GENERATING INTENTION POINTS

Distribution of Intention Points	minADE ↓	Miss Rate ↓	**mAP ↑**
uniform grids	0.7022	0.1952	0.3205
k-means clustering	**0.6557**	**0.1575**	**0.3539**

strategies, where our proposed learnable intention query demonstrates significantly superior results. Specifically, our strategy achieves a much better mAP compared to the previous latent anchor embedding [50] (i.e., +5.53%) and dense-goal-based methods [21], [67] (i.e., +4.67%). This improvement can be attributed to our mode-specific querying strategy, where each intention query is associated with an explicit intention point, enabling more accurate and precise multimodal predictions.

Effects of the Distribution of Intention Points: As introduced in Section III-B, we utilize the k-means clustering algorithm to generate 64 intention points, which serve as the foundation for our intention queries. In order to compare this approach with the straightforward uniform sampling strategy, we uniformly sample $8\times8 = 64$ intention points by considering the range of trajectory distribution for each category (see Fig. 3). The results presented in Table VI indicate a significant drop in performance when replacing the k-means clustering algorithm with uniform sampling for generating intention points. This comparison highlights the superiority of our k-means clustering algorithm, as it produces a more accurate and comprehensive distribution of intention points. Consequently, it effectively captures the diverse future motion intentions of our interested agent with a small number of intention points.

Effects of the Iterative Trajectory Refinement: In Section III-A, we introduce the utilization of stacked transformer decoder layers for iterative refinement of predicted trajectories by continually aggregating fine-grained features via dynamic map collection. As shown in the last two rows of Table V, this iterative refinement approach significantly reduces the miss rate metric by 1.48% and improves the performance of mAP by +1.6%. By continually aggregating trajectory-specific features from the context encoder with the proposed intention queries, the refinement process effectively improves the accuracy and quality of the predicted trajectories.

Effects of Local Attention for Context Encoding: Table VII demonstrates that the utilization of local self-attention in our context encoders leads to slightly superior performance compared to global attention when using the same number of map polylines as input. This finding confirms the significance of incorporating the input's local structure for more effective context encoding, and the inclusion of such prior knowledge through local attention positively impacts performance. Moreover, local attention proves to be more memory-efficient, allowing for performance improvements even when increasing the number of map polylines from 256 to 1,024. In contrast, global attention suffers from memory limitations due to its quadratic complexity.

TABLE VII
EFFECTS OF LOCAL SELF-ATTENTION IN THE TRANSFORMER ENCODER OF THE MTR FRAMEWORK

Attention	#Polyline	minADE ↓	Miss Rate ↓	**mAP ↑**
Global	256	0.6701	0.1623	0.3450
Global	512	0.6677	0.1610	0.3495
Global	768	OOM	OOM	OOM
Local	256	0.6692	0.1633	0.3522
Local	512	0.6685	0.1599	0.3515
Local	768	**0.6557**	0.1575	0.3539
Local	1024	0.6601	**0.1555**	**0.3564**

"#polyline" is the number of input map polylines used for context encoding. "OOM" indicates running out of memory.

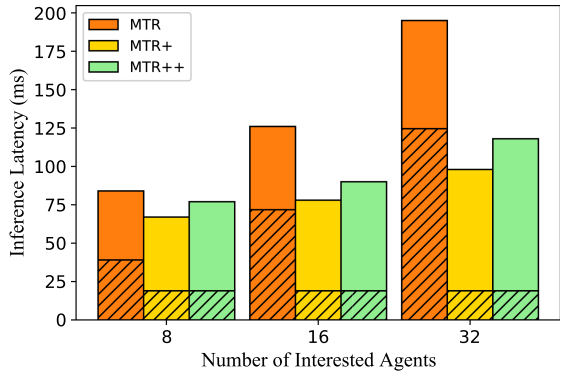


Fig. 8. Comparison of inference latency across different numbers of focal agents (i.e., interested agents) required to predict their future trajectories. The hatched area at the bottom of each pillar indicates the inference latency of the corresponding context encoder. MTR+ indicates the results obtained by only incorporating the symmetric context encoder into the MTR framework, while MTR++ indicates the results by further incorporating the mutually-guided intention querying strategy.

Effects of the Symmetric Scene Context Modeling Module: In Section IV-A, we present the symmetric scene context encoding module, which utilizes a shared context encoder for motion prediction of multiple interested agents in the same scene. Table VIII demonstrates the effectiveness of incorporating our symmetric context encoder into the MTR framework (referred to as MTR+). With MTR+, we achieve comparable performance to the MTR framework while significantly reducing both inference latency and memory cost. Specifically, when the number of interested agents increases from 8 to 32, MTR requires individual scene context encoding for each agent, causing a substantial increase in inference latency and memory cost. In contrast, MTR+ utilizes the query-centric self-attention module to encode the entire scene with a shared symmetric context encoder, leading to a remarkable reduction in inference latency (from 193 ms to 98 ms for 32 interested agents) and memory cost (from 15.6 GB to 4.7 GB for 32 interested agents). Furthermore, we provide a breakdown analysis of inference latency in Fig. 8, which demonstrates that as the number of interested agents increases, the latency of MTR's context encoder significantly rises, while the latency of MTR+'s context encoder remains constant due

TABLE VIII
EFFECTS OF THE SYMMETRIC SCENE CONTEXT ENCODER AND MUTUAL-GUIDED INTENTION QUERY IN THE MTR++ FRAMEWORK

Method	Symmetric Context Encoder	Mutually-Guided Intention Querying	Performance with the Given Focal Agents				Efficiency with Increasing Numbers of Focal Agents					
							Inference Latency			Memory Cost		
			minADE ↓	minFDE ↓	Miss Rate ↓	**mAP ↑**	8	16	32	8	16	32
MTR			0.6557	1.3362	0.1575	0.3539	84ms	123ms	193ms	5.2GB	7.1GB	15.6GB
MTR+	✓		0.6679	1.3486	0.1588	0.3505	67ms	78ms	98ms	2.9GB	3.2GB	4.7GB
MTR++	✓	✓	**0.6490**	**1.3163**	**0.1559**	**0.3754**	77ms	90ms	118ms	3.1GB	3.4GB	5.2GB

The performance evaluation is conducted using a maximum of 8 given focal agents from the validation set of WOMD. To further assess the efficiency of different models as the number of focal agents increases, we set 8, 16, and 32 fake agents as the focal agents for calculating their inference efficiency and memory efficiency.

TABLE IX
EFFECTS OF THE INTERACTION OF INTENTION QUERIES IN THE MTR++ FRAMEWORK

Interaction of Intention Queries		minADE ↓	Miss Rate ↓	**mAP ↑**
Within Each Agent	Across Different Agents			
		0.6708	0.1625	0.3484
✓		0.6679	0.1588	0.3505
	✓	0.6624	0.1566	0.3541
✓	✓	**0.6490**	**0.1559**	**0.3754**

"Within Each Agent" and "Across Different Agents" indicates enabling the information interaction within the intention queries of each agent and across the intention queries of different agents, respectively.

TABLE X
EFFECTS OF THE QUERY-CENTRIC SELF-ATTENTION MODULE IN THE MTR++ FRAMEWORK

Positional Encoding in Self-Attention			minADE ↓	Miss Rate ↓	**mAP ↑**
Strategy	With Query/Key	With Value			
None			0.6886	0.1614	0.3513
Global	✓	✓	0.6913	0.1627	0.3463
Query-Centric	✓	✓	**0.6490**	**0.1559**	**0.3754**
Query-Centric	✓		0.6523	0.1570	0.3658
Query-Centric		✓	0.6814	0.1603	0.3574

"With Query/Key" and "With Value" indicates adding the position embedding to query/key tokens and value tokens, respectively.

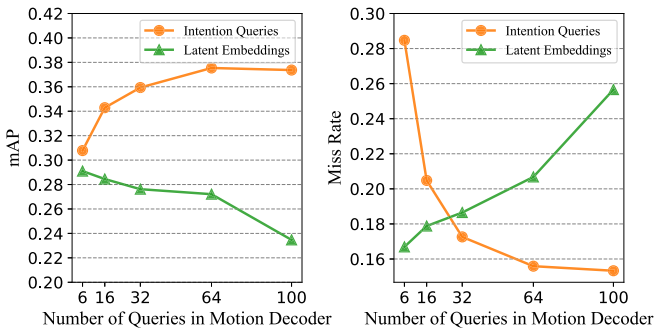


Fig. 9. Comparison of explicit intention queries and implicit latent embeddings in terms of different numbers of queries in the MTR++ framework. Two different colored curves demonstrate the performance of different strategies for generating future trajectories.

to the utilization of shared context features, since these shared context features enable the prediction of future trajectories for any number of agents within the scene.

Effects of the Mutually-Guided Intention Querying Strategy: Building upon our proposed symmetric scene context encoder for joint motion prediction, we introduce the mutually-guided intention querying strategy in SectionIV-B. This strategy enables the interaction of future behaviors among multiple agents through the propagation of information among their intention queries. In Table VIII, we observe that the mutually-guided intention querying strategy significantly enhances the performance of MTR+ with a remarkable mAP improvement of $+2.49\%$. This improvement demonstrates the effectiveness of broadcasting the potential future behaviors of each agent to other agents via their intention queries, allowing MTR++ to predict more confident future behaviors by considering the overall development of the scene elements.

Furthermore, as each agent incorporates multiple intention queries (i.e., 64 in MTR frameworks), we investigate the interaction among these intention queries within each agent and across different agents. As presented in Table IX, removing either type of interaction results in a significant decrease in performance by at least -2.13% in terms of mAP. Removing both types of interaction leads to a larger performance drop of -2.70% in terms of mAP. This analysis highlights the importance of the interaction among an agent's different intention queries, enabling the generation of more accurate multimodal future trajectories. Additionally, the interaction among intention queries across different agents empowers the model to predict informed and scene-compliant future trajectories for multiple agents, thereby yielding additional performance improvement.

Effects of the Query-Centric Self-Attention: We introduce the query-centric self-attention module in Section IV-A, which plays a vital role in modeling the relationship between tokens within their respective local coordinate systems. This module enables both symmetric scene context modeling and mutually-guided intention querying. In Table X, we examine the effects of different positional encoding strategies in query-centric self-attention. The results in the first three rows indicate that query-centric relative positional encoding is crucial for achieving optimal performance. Removing this encoding or replacing it with global positional encoding significantly decreases performance by -2.41% and -2.91% in terms of mAP, respectively. This finding demonstrates the importance of modeling the relationship in the local coordinate system of each query token, as it benefits the simultaneous motion prediction for multiple agents by treating all tokens symmetrically. Additionally, comparing the last three rows of Table X, we observe that adding positional embeddings to both the query/key tokens and value tokens yields the best performance.

Effects of the Number of Intention Queries: We conduct an ablation study to investigate the impact of the number of intention queries on the performance of the MTR++ framework. In Fig. 9, we vary the number of intention queries by generating their intention points using the k-means clustering algorithm on the training dataset. The orange curves in Fig. 9 illustrate that the performance of the MTR++ framework improves significantly as the number of intention queries increases from 6

TABLE XI
EFFECTS OF THE DENSE FUTURE PREDICTION MODULE IN THE MTR++ FRAMEWORK

Dense Future Prediction	minADE ↓	Miss Rate ↓	**mAP ↑**
	0.6662	0.1639	0.3606
✓	**0.6490**	**0.1559**	**0.3754**

to 64. However, the performance saturates when the number of intention queries is further increased to 100. This ablation experiment highlights that incorporating 64 intention queries in the MTR frameworks already enables the coverage of diverse and wide-ranging future trajectories. This achievement is attributed to the design of learnable intention queries, which proves to be more efficient compared to previous goal-based strategies [21], [67] that require a large number of goal candidates to achieve satisfactory performance.

Discussion of Explicit Intention Queries and Implicit Latent Embeddings: In comparison to the latent anchor embeddings proposed in the state-of-the-art work MultiPath++[50], our proposed MTR frameworks establish a direct correspondence between intention queries and motion modes. This explicit mapping leads to faster convergence and improved performance. By referring to Fig. 9, we can observe the following findings regarding the comparison between intention queries and latent embeddings with varying numbers of queries for the motion decoder: (1) Our strategy outperforms latent embeddings in terms of mAP and miss rate as the number of queries increases. This improvement is attributed to the fact that each intention query is specifically assigned to a particular motion mode, enabling a more stable training process. Conversely, in the case of latent embeddings, a ground truth trajectory can randomly associate with different anchor embeddings during training due to the lack of explicit correspondence. This randomness leads to training instability and decreased performance when increasing the number of anchor embeddings. (2) The explicit semantic interpretation of each intention query also contributes to its superior performance in terms of mAP. Intention queries are capable of predicting trajectories with more confident scores, thereby positively influencing the mAP metric. Overall, the establishment of explicit correspondence between intention queries and motion modes in our approach results in faster convergence, enhanced stability, and improved performance compared to previous latent embeddings.

Effects of Dense Future Prediction: We investigate the impact of the dense future prediction module in Table XI. By removing this module, we observe a significant decrease in the performance of the MTR++ framework, with a -1.48% drop in mAP. We attribute this result to the beneficial effects of the dense future prediction module. It not only provides dense supervision for the context encoder, enabling it to learn more effective features for motion prediction of all agents in the scene, but also enhances the motion decoding process in the decoder network by incorporating agent features with their potential future trajectories, thereby enriching the contextual information for multimodal motion prediction.

Effects of the Number of Decoder Layers: We investigate the number of transformer decoder layers in the MTR++ framework in Table XII. We observe a consistent improvement in performance as we increase the number of decoder layers from 1 to 6. This improvement can be attributed to the stacked transformer decoder layers with the mutually-guided intention querying module, which facilitates the generation of more scene-compliant future trajectories through iterative trajectory refinement based on the predicted behaviors of other agents. However, increasing the number of decoder layers to 9 does not yield further improvement, suggesting a diminishing return. As a result, we adopt 6 decoder layers in our MTR++ framework to strike a balance between performance and efficiency.

TABLE XII
EFFECTS OF THE NUMBER OF DECODER LAYERS IN THE MTR++ FRAMEWORK

Number of Decoder Layers	minADE ↓	Miss Rate ↓	**mAP ↑**
1	0.6882	0.1689	0.3274
2	0.6665	0.1636	0.3561
3	0.6641	0.1653	0.3699
6	**0.6490**	**0.1559**	**0.3754**
9	0.6512	0.1573	0.3728

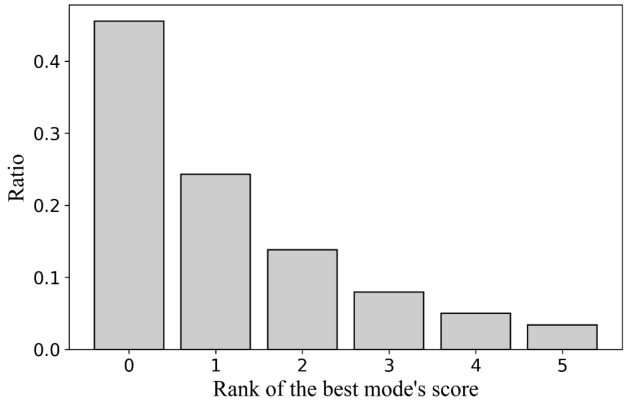
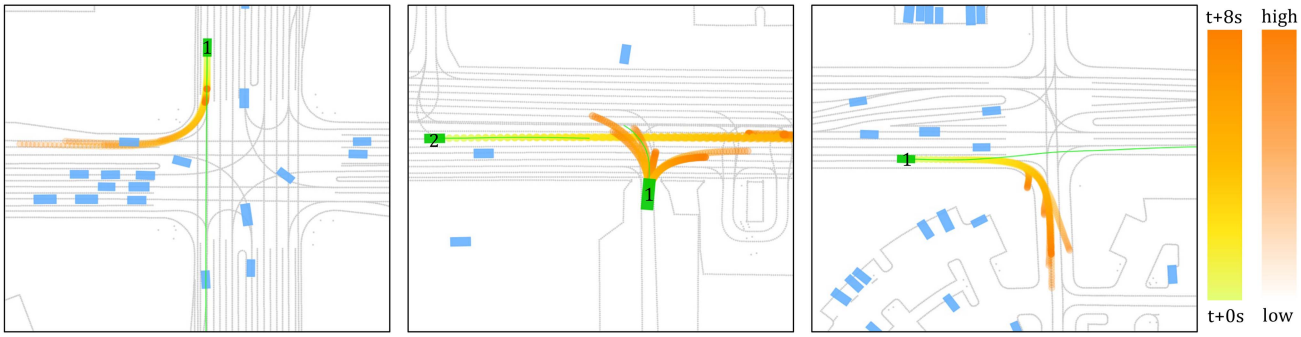


Fig. 10. Rank distribution of the best mode's score of our MTR++ framework on the Waymo Open Motion Dataset.

D. Discussion of Failure Cases and Future Challenges

While MTR/MTR++ frameworks have demonstrated remarkable performance and achieved a state-of-the-art position, they continue to confront two primary challenges as below.

Quality Score Estimation: As evident in Table I, our model achieves a relatively lower average precision (i.e., below 0.5), while maintaining a favorable miss rate (approximately 0.13). This discrepancy arises due to the inadequacy of current quality score estimations in accurately reflecting the quality of each predicted trajectory, resulting in significant penalties imposed by the average precision metric. A compelling illustration of this issue is provided in Fig. 10, where our analysis reveals that only 45.5% of agents adhere to the predicted trajectory with the highest quality score. This underscores the limitations of existing quality score predictions, ultimately contributing to suboptimal average precision. Consequently, one important direction for future research involves investigating methods to refine the alignment between predicted scores and the quality of the corresponding trajectory. This would yield a more precise distribution of an agent's future behaviors, enhancing the predictive capabilities of our models.



(a) **V1** is moving forward to pass the intersection, but the model only predicts turning right behaviors.

(b) **V2** will yield **V1** in the near future, but the model predicts that **V1** will yield **V2**.

(c) Although **V1** is on the right turn lane, it will move forward finally. The model's predictions do not include this behavior.

Fig. 11. Qualitative results of the failure cases of MTR frameworks on WOMD. The green curves indicate the ground-truth trajectories of our interested agents (green rectangle). We use gradient color to visualize the predicted trajectory waypoints at different future time steps, and trajectory confidence is visualized by setting different transparent. Abbreviation: Vehicle (V).

Diverse and Accurate Multimodal Behaviors in Rare Scenarios: Another significant challenge centers on generating diverse and accurate multimodal behaviors that faithfully capture the real intentions of the agent, particularly in rare scenarios. As illustrated in Fig. 11, our model, in certain scenarios, may generate homogenized future trajectories that fail to encompass the genuine intentions of the agent, resulting in imprecise multimodal predictions. Thus, a pivotal future research endeavor involves the development of methods that yield comprehensive multimodal behaviors, ensuring that multiple future trajectories not only exist but also represent distinct, meaningful intentions of the agent rather than merely variations along the same direction.

These challenges and future research directions are instrumental in advancing the capabilities of motion prediction models, and addressing them will contribute to more accurate and comprehensive predictions, ultimately benefitting a wide range of applications.

VI. Conclusion

In this paper, we have introduced the Motion TRansformer (MTR) frameworks as novel solutions for motion prediction in autonomous driving systems. The MTR frameworks employ a transformer encoder-decoder structure with learnable intention queries, effectively combining global intention localization and local movement refinement processes. This design enables the accurate determination of the agent's intent and adaptive refinement of predicted trajectories, resulting in efficient and precise prediction of multimodal future trajectories. Moreover, the proposed MTR++ framework enhances these capabilities by incorporating symmetric scene context modeling and mutually-guided intention querying modules, enabling the prediction of multimodal motion for multiple agents in a scene-compliant manner. Experimental results on the large-scale WOMD dataset demonstrate the state-of-the-art performance of the MTR frameworks on both marginal and joint motion prediction benchmarks.

References

[1] Argoverse 2, "Argoverse 2: Motion forecasting competition," 2022. Accessed: Aug. 02, 2022. [Online]. Available: https://eval.ai/web/challenges/challenge-page/1719/leaderboard/4098

[2] A. Alahi, K. Goel, V. Ramanathan, A. Robicquet, L. Fei-Fei, and S. Savarese, "Social LSTM: Human trajectory prediction in crowded spaces," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 961–971.

[3] H. Bao, L. Dong, and F. Wei, "BEiT: BERT pre-training of image transformers," in *Proc. Int. Conf. Learn. Representations*, 2022.

[4] Y. Biktairov, M. Stebelev, I. Rudenko, O. Shliazhko, and B. Yangel, "PRANK: Motion prediction based on ranking," in *Proc. 34th Int. Conf. Neural Inf. Process. Syst.*, 2020, Art. no. 215.

[5] N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, and S. Zagoruyko, "End-to-end object detection with transformers," in *Proc. Eur. Conf. Comput. Vis.*, 2020, pp. 213–229.

[6] S. Casas, C. Gulino, R. Liao, and R. Urtasun, "SpAGNN: Spatially-aware graph neural networks for relational behavior forecasting from sensor data," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2020, pp. 9491–9497.

[7] S. Casas, C. Gulino, S. Suo, K. Luo, R. Liao, and R. Urtasun, "Implicit latent variable model for scene-consistent motion forecasting," in *Proc. Eur. Conf. Comput. Vis.*, 2020, pp. 624–641.

[8] S. Casas, W. Luo, and R. Urtasun, "IntentNet: Learning to predict intention from raw sensor data," in *Proc. 2nd Conf. Robot Learn.*, 2018, pp. 947–956.

[9] S. Casas, A. Sadat, and R. Urtasun, "MP3: A unified model to map, perceive, predict and plan," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2021, pp. 14398–14407.

[10] Y. Chai, B. Sapp, M. Bansal, and D. Anguelov, "MultiPath: Multiple probabilistic anchor trajectory hypotheses for behavior prediction," in *Proc. Conf. Robot Learn.*, 2019, pp. 86–99.

[11] X. Chen, S. Shi, B. Zhu, K. C. Cheung, H. Xu, and H. Li, "MPPNet: Multi-frame feature intertwining with proxy points for 3D temporal object detection," in *Proc. Eur. Conf. Comput. Vis.*, 2022, pp. 680–697.

[12] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," 2018, *arXiv:1810.04805*.

[13] N. Djuric et al., "Uncertainty-aware short-term motion prediction of traffic actors for autonomous driving," in *Proc. IEEE Winter Conf. Appl. Comput. Vis.*, 2020, pp. 2084–2093.

[14] A. Dosovitskiy et al., "An image is worth 16x16 words: Transformers for image recognition at scale," in *Proc. Int. Conf. Learn. Representations*, 2021.

[15] 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.

[16] L. Fang, Q. Jiang, J. Shi, and B. Zhou, "TPNet: Trajectory proposal network for motion prediction," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 6796–6805.

[17] 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.

[18] T. Gilles, S. Sabatini, D. Tsishkou, B. Stanciulescu, and F. Moutarde, "GOHOME: Graph-oriented heatmap output for future motion estimation," 2021, *arXiv:2109.01827*.

[19] T. Gilles, S. Sabatini, D. Tsishkou, B. Stanciulescu, and F. Moutarde, "HOME: Heatmap output for future motion estimation," in *Proc. IEEE Int. Intell. Transp. Syst. Conf.*, 2021, pp. 500–507.

[20] T. Gilles, S. Sabatini, D. Tsishkou, B. Stanciulescu, and F. Moutarde, "THOMAS: Trajectory heatmap output with learned multi-agent sampling," 2021, *arXiv:2110.06607*.

[21] J. Gu, C. Sun, and H. Zhao, "DenseTNT: End-to-end trajectory prediction from dense goal sets," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2021, pp. 15283–15292.

[22] A. Gupta, J. Johnson, L. Fei-Fei, S. Savarese, and A. Alahi, "Social GAN: Socially acceptable trajectories with generative adversarial networks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 2255–2264.

[23] J. Hong, B. Sapp, and J. Philbin, "Rules of the road: Predicting driving behavior with a convolutional model of semantic interactions," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 8446–8454.

[24] X. Jia et al., "Towards capturing the temporal dynamics for trajectory prediction: A coarse-to-fine approach," in *Proc. 6th Conf. Robot Learn.*, 2023, pp. 910–920.

[25] X. Jia, L. Sun, M. Tomizuka, and W. Zhan, "IDE-net: Interactive driving event and pattern extraction from human data," *IEEE Robot. Autom. Lett.*, vol. 6, no. 2, pp. 3065–3072, Apr. 2021.

[26] X. Jia, L. Sun, H. Zhao, M. Tomizuka, and W. Zhan, "Multi-agent trajectory prediction by combining egocentric and allocentric views," in *Proc. 5th Conf. Robot Learn.*, 2022, pp. 1434–1443.

[27] X. Jia, P. Wu, L. Chen, H. Li, Y. Liu, and J. Yan, "HDGT: Heterogeneous driving graph transformer for multi-agent trajectory prediction via scene encoding," 2022, *arXiv:2205.09753*.

[28] S. Konev, K. Brodt, and A. Sanakoyeu, "MotionCNN: A strong baseline for motion prediction in autonomous driving," in *Proc. Conf. Comput. Vis. Pattern Recognit. Workshop*, 2021, pp. 1–6.

[29] X. Lai et al., "Stratified transformer for 3D point cloud segmentation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2022, pp. 8490–8499.

[30] J. Li, F. Yang, M. Tomizuka, and C. Choi, "EvolveGraph: Multi-agent trajectory prediction with dynamic relational reasoning," in *Proc. 34th Int. Conf. Neural Inf. Process. Syst.*, 2020, Art. no. 1660.

[31] M. Liang et al., "Learning lane graph representations for motion forecasting," in *Proc. Eur. Conf. Comput. Vis.*, 2020, pp. 541–556.

[32] S. Liu et al., "DAB-DETR: Dynamic anchor boxes are better queries for DETR," in *Proc. Int. Conf. Learn. Representations*, 2022.

[33] Y. Liu, J. Zhang, L. Fang, Q. Jiang, and B. Zhou, "Multimodal motion prediction with stacked transformers," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2021, pp. 7573–7582.

[34] K. Mangalam et al., "It is not the journey but the destination: Endpoint conditioned trajectory prediction," in *Proc. Eur. Conf. Comput. Vis.*, 2020, pp. 759–776.

[35] F. Marchetti, F. Becattini, L. Seidenari, and A. Del Bimbo, "MANTRA: Memory augmented networks for multiple trajectory prediction," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 7141–7150.

[36] D. Meng et al., "Conditional DETR for fast training convergence," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2021, pp. 3631–3640.

[37] J. Mercat, T. Gilles, N. El Zoghby, G. Sandou, D. Beauvois, and G. Pita Gil, "Multi-head attention for multi-modal joint vehicle motion forecasting," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2020, pp. 9638–9644.

[38] X. Mo, Z. Huang, and C. Lv, "Multi-modal interactive agent trajectory prediction using heterogeneous edge-enhanced graph attention network," in *Proc. Conf. Comput. Vis. Pattern Recognit. Workshop*, 2021, pp. 1–7.

[39] J. Ngiam et al., "Scene transformer: A unified architecture for predicting future trajectories of multiple agents," in *Proc. Int. Conf. Learn. Representations*, 2022.

[40] S. H. Park et al., "Diverse and admissible trajectory forecasting through multimodal context understanding," in *Proc. Eur. Conf. Comput. Vis.*, 2020, pp. 282–298.

[41] T. Phan-Minh, E. C. Grigore, F. A. Boulton, O. Beijbom, and E. M. Wolff, "CoverNet: Multimodal behavior prediction using trajectory sets," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 14062–14071.

[42] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, "PointNet: Deep learning on point sets for 3D classification and segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 77–85.

[43] N. Rhinehart, K. M. Kitani, and P. Vernaza, "R2P2: A reparameterized pushforward policy for diverse, precise generative path forecasting," in *Proc. Eur. Conf. Comput. Vis.*, 2018, pp. 794–811.

[44] N. Rhinehart, R. McAllister, K. Kitani, and S. Levine, "PRECOG: PREdiction conditioned on goals in visual multi-agent settings," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2019, pp. 2821–2830.

[45] T. Salzmann, B. Ivanovic, P. Chakravarty, and M. Pavone, "Trajectron: Dynamically-feasible trajectory forecasting with heterogeneous data," in *Proc. Eur. Conf. Comput. Vis.*, 2020, pp. 683–700.

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

[47] Q. Sun, X. Huang, J. Gu, B. C. Williams, and H. Zhao, "M2I: From factored marginal trajectory prediction to interactive prediction," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2022, pp. 6533–6542.

[48] C. Tang and R. R. Salakhutdinov, "Multiple futures prediction," in *Proc. 33rd Int. Conf. Neural Inf. Process. Syst.*, 2019, Art. no. 1382.

[49] E. Tolstaya, R. Mahjourian, C. Downey, B. Vadarajan, B. Sapp, and D. Anguelov, "Identifying driver interactions via conditional behavior prediction," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2021, pp. 3473–3479.

[50] B. Varadarajan et al., "MultiPath: Efficient information fusion and trajectory aggregation for behavior prediction," in *Proc. Int. Conf. Robot. Autom.*, 2022, pp. 7814–7821.

[51] A. Vaswani et al., "Attention is all you need," in *Proc. 31st Int. Conf. Neural Inf. Process. Syst.*, 2017, pp. 6000–6010.

[52] H. Wang et al., "DSVT: Dynamic sparse voxel transformer with rotated sets," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2023, pp. 13520–13529.

[53] J. Wang, H. Xu, M. Narasimhan, and X. Wang, "Multi-person 3D motion prediction with multi-range transformers," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2021, pp. 6036–6049.

[54] X. Wang, R. Girshick, A. Gupta, and K. He, "Non-local neural networks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 7794–7803.

[55] Y. Wang et al., "TENET: Transformer encoding network for effective temporal flow on motion prediction," 2022, *arXiv:2207.00170*.

[56] Waymo, "Waymo open dataset interaction prediction challenge 2021," 2021. Accessed: May 25, 2023. [Online]. Available: https://waymo.com/open/challenges/2021/interaction-prediction/

[57] Waymo, "Waymo open dataset motion prediction challenge 2022," 2022. Accessed: May 25, 2023. [Online]. Available: https://waymo.com/open/challenges/2022/motion-prediction/

[58] Waymo, "Waymo open dataset motion prediction challenge 2023," 2023. Accessed: May 25, 2023. [Online]. Available: https://waymo.com/open/challenges/2023/motion-prediction/

[59] B. Wilson et al., "Argoverse 2: Next generation datasets for self-driving perception and forecasting," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2021.

[60] D. Wu and Y. Wu, "Air2 for interaction prediction," in *Proc. Conf. Comput. Vis. Pattern Recognit. Workshop*, 2021, pp. 1–6.

[61] Y. Xu, D. Ren, M. Li, Y. Chen, M. Fan, and H. Xia, "Tra2Tra: Trajectory-to-trajectory prediction with a global social spatial-temporal attentive neural network," *IEEE Robot. Autom. Lett.*, vol. 6, no. 2, pp. 1574–1581, Apr. 2021.

[62] Z. Yang, L. Jiang, Y. Sun, B. Schiele, and J. Jia, "A unified query-based paradigm for point cloud understanding," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2022, pp. 8531–8541.

[63] M. Ye, T. Cao, and Q. Chen, "TPCN: Temporal point cloud networks for motion forecasting," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2021, pp. 11313–11322.

[64] F. Zeng, B. Dong, T. Wang, X. Zhang, and Y. Wei, "MOTR: End-to-end multiple-object tracking with transformer," 2021, *arXiv:2105.03247*.

[65] H. Zhang et al., "DINO: DETR with improved denoising anchor boxes for end-to-end object detection," 2022, *arXiv:2203.03605*.

[66] Y. Zhang, J. Zhang, J. Zhang, J. Wang, K. Lu, and J. Hong, "A novel learning framework for sampling-based motion planning in autonomous driving," in *Proc. 34th AAAI Conf. Artif. Intell.*, 2020, pp. 1202–1209.

[67] H. Zhao et al., "TNT: Target-driven trajectory prediction," in *Proc. Conf. Robot Learn.*, 2020, pp. 895–904.

[68] C. Lv, Z. Huang, and X. Mo, "ReCoAt: A deep learning framework with attention mechanism for multi-modal motion prediction," in *Proc. Conf. Comput. Vis. Pattern Recognit. Workshop*, 2021, pp. 1–6.

[69] Z. Zhou, L. Ye, J. Wang, K. Wu, and K. Lu, "HiVT: Hierarchical vector transformer for multi-agent motion prediction," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2022, pp. 8813–8823.

[70] X. Zhu, W. Su, L. Lu, B. Li, X. Wang, and J. Dai, "Deformable DETR: Deformable transformers for end-to-end object detection," in *Proc. Int. Conf. Learn. Representations*, 2021.
[71] Y. Zhu, D. Qian, D. Ren, and H. Xia, "StarNet: Pedestrian trajectory prediction using deep neural network in star topology," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2019, pp. 8075–8080.

Shaoshuai Shi received the PhD degree from the Department of Electronic Engineering, The Chinese University of Hong Kong, in 2021. He is currently a postdoctoral researcher with the Department of Computer Vision and Machine Learning of Max Planck Institute for Informatics. His research focuses on computer vision and machine learning, particularly in 3D scene understanding, object detection, motion prediction, knowledge transfer, as well as their applications in autonomous driving and robotics.

Li Jiang received the PhD degree from the Department of Computer Science and Engineering, The Chinese University of Hong Kong, in 2021. She is currently an assistant professor with The Chinese University of Hong Kong, Shenzhen. Before that, She was a postdoctoral researcher with the Department of Computer Vision and Machine Learning of Max Planck Institute for Informatics. Her research interest includes computer vision and deep learning, particularly in 3D scene understanding, efficient representation learning, autonomous driving, and robotics.

Dengxin Dai received the PhD degree in computer vision from ETH Zurich, Switzerland, in 2016. He was a senior researcher with MPI for Informatics and a lecturer (external) with ETH Zurich. His research interests include autonomous driving, robust perception in adverse weather and illumination conditions, automotive sensors and computer vision under limited supervision. He has organized a CVPR Workshop series ('19, '20) on Vision for All Seasons: Bad Weather and Nighttime, and has organized an ICCV'19 workshop on Autonomous Driving. He has been a program committee member of several major computer vision conferences and received multiple outstanding reviewer awards. He is also a guest editor for *International Journal of Computer Vision* and the area chair for WACV'20 and CVPR'21.

Bernt Schiele (Fellow, IEEE) received the master's degrees both from the University of Karlsruhe, Germany, and ENSIMAG, Grenoble, France, in 1994, and the PhD degree from INP Grenoble, France, in 1997. In 1994 he was a visiting researcher with Carnegie Mellon University, Pittsburgh. From 1999 until 2004, he was an assistant professor with ETH Zurich and, from 2004 to 2010, he was a full professor of computer science with TU Darmstadt. In 2010, he was appointed a scientific member of the Max Planck Society and a director with the Max Planck Institute for Informatics. Since 2010, he has also been a professor with Saarland University. He is a fellow of ACM, ELLIS, and IAPR.