Proc of the 2025 IEEE International
Conference on Real-time Computing and Robotics
June 1 - 6, 2025, Toyama, Japan

2025 IEEE International Conference on Real-time Computing and Robotics (RCAR) | 979-8-3315-0205-8/25/$31.00 ©2025 IEEE | DOI: 10.1109/RCAR65431.2025.11139566

Uncertainty-Aware Module for Trajectory Prediction and Ego-Planning of Challenging Agents in Autonomous Driving

Yuheng Chen, Yesheng Zhang and Xu Zhao*

***Abstract*— In autonomous driving, trajectory prediction requires modeling the interactions between agents and maps, hence enhancing the ego-planning. Most methods adopt an encoder-decoder framework, where the encoder extracts features and the multi-layer decoder predicts trajectories. However, these approaches typically treat all agents equally, overlooking the varying levels of prediction difficulty among different agents. To address this limitation, we propose a Uncertainty-Aware Module (UAM) to capture and quantify the prediction uncertainty for each agent. By incorporating uncertainty-aware decoding, the module fixes the trajectories of agents with lower prediction uncertainty early, allowing the model to focus on more challenging agents. This approach not only improves both accuracy and computational efficiency but can also be seamlessly integrated as a plug-and-play module into state-of-the-art networks. By adding UAM to Gameformer, we outperform Gameformer in both accuracy and efficiency on nuPlan Dataset. Additionally, we propose a Mixture of Experts planning framework to integrate rule-based method and Gameformer with UAM. Experiments on nuPlan show state-of-the-art performance.**

I. INTRODUCTION

Autonomous driving is a rapidly developing field, where accurate trajectory prediction is crucial for safe and efficient vehicle planning. One of the key challenges in trajectory prediction is modeling the complex interactions between multiple agents and the surrounding environment. Due to the inherent uncertainty of agents, this process necessitates multimodal trajectory prediction to account for a range of possible future behaviors, as relying on a single trajectory often fails to capture the full spectrum of potential outcomes. To this end, recent methods have employed deep learning techniques, particularly in the form of Transformer encoder and decoder architecture, where the encoder extracts spatial and temporal features from the agent history and the environment, and the decoder predicts the future trajectories of agents. In Transformer decoder, multiple decoding layers are typically stacked, with each layer passing query content to the next. Additionally, multimodal predicted trajectories are obtained through a Gaussian Mixture Model (GMM) module integrated within the decoding process, such as MTR [1] and Gameformer [2].

Despite the success of such approaches, a significant limitation remains: these models typically treat all agents with the same level of importance, failing to account for the varying difficulty in predicting the trajectories of different agents. Agents with simpler, more predictable behaviors are treated with the same level of attention as those exhibiting more complex, unpredictable motions, which leads to suboptimal prediction performance and inefficient resource utilization.

To address this challenge, we propose a Uncertainty-Aware Module (UAM) that quantifies the prediction uncertainty of each agent, enabling the model to allocate resources more effectively. UAM operates by analyzing the multimodal predicted trajectories generated at each layer of the decoder, along with the scores assigned to each predicted modality. The distribution of different modal trajectories and their corresponding scores reflects the uncertainty of the prediction. For agents with lower prediction uncertainty, the trajectories are typically concentrated around a few modes, or one trajectory's score significantly surpasses the others. Conversely, agents with higher prediction uncertainty exhibit more dispersed multimodal trajectory distributions, with scores that are relatively close to each other. By modeling the divergence of these trajectory distributions and normalizing the scores, UAM provides a robust measure of prediction uncertainty. By incorporating this uncertainty-aware decoding, our model is able to focus its attention on more challenging agents, fixing the trajectories of simpler agents early in the prediction process. This approach not only improves prediction accuracy but also enhances computational efficiency. Moreover, UAM can be integrated as a plug-and-play module with state-of-the-art trajectory prediction models, such as Gameformer, leading to improved performance in terms of both accuracy and efficiency.

We further extend this idea with a Mixture of Experts (MoE) planning framework that integrates a rule-based method with Gameformer enhanced by UAM. Experiments conducted on the nuPlan [3] dataset demonstrate the state-of-the-art performance of our approach, which outperforms existing models in both accuracy and efficiency.

In summary, the contributions of this paper are as follows:

1) **Uncertainty-Aware Module (UAM)**. We propose a novel Uncertainty-Aware Module (UAM) that quantifies the prediction uncertainty of each agent by modeling the divergence of multimodal trajectory distributions. This module enables uncertainty-aware decoding to allocate computational resources more effectively and improve trajectory prediction and ego-planning. UAM can be seamlessly integrated as a plug-and-play module into existing models like Gameformer, leading to enhanced performance in terms of both accuracy and efficiency.
2) **Mixture of Experts (MoE) planning framework**. We

Yuheng Chen, Yesheng Zhang, and Xu Zhao are with the department of Automation, Laboratory of Computer Vision, Shanghai Jiao Tong University, Shanghai 200240, China
*Corresponding Author. Email: zhaoxu@sjtu.edu.cn

develop a planning framework that combines a rule-based method with Gameformer enhanced by UAM. This hybrid framework leverages the strengths of both methods to achieve improved planning performance in autonomous driving scenarios.

3) **State-of-the-art performance on the nuPlan dataset**. Through extensive experiments on the nuPlan dataset, we demonstrate that our method achieves state-of-the-art performance in trajectory prediction and planning (include open loop planning, closed loop nonreactive simulation and closed loop reactive simulation), outperforming existing models.

II. Related Work

A. Multimodal Trajectory Prediction

Multimodal trajectory prediction is crucial for autonomous driving, given the inherent uncertainty of agents' future behaviors. Early methods, leveraging CNNs in computer vision, converted trajectory inputs and maps into rasterized images, where trajectories were marked on grid maps. However, this approach introduced significant redundancy, leading to high computational costs.

VectorNet [4] was the first to use vectorized representations for trajectory prediction, greatly reducing computational overhead. TNT [5] treats trajectory prediction as a target-driven problem, dividing it into three steps: target point prediction, trajectory estimation based on the target point, and trajectory scoring and selection. DenseTNT [6] builds on this by replacing sparse anchor-based sampling with dense sampling for improved accuracy. With the advent of Transformers, encoder-decoder architectures have become popular in trajectory prediction. For instance, SceneTransformer [7] models interactions between agents, overcoming the limitations of independent trajectory predictions. HPTR [8] designs a hierarchical Transformer encoder to better encode historical trajectories and map information. On the decoder side, MTR [1] introduces modules for global intent localization and local motion refinement, with specific queries assigned to different responsibilities.

Despite these advancements, most methods do not explicitly model the complex interactions between agents' future trajectories. Gameformer addresses this gap by incorporating a K-layer game-theoretic structure in the decoder, explicitly modeling these interactions. However, it does not account for varying prediction uncertainty across agents. To address this, we propose the Uncertainty-Aware Module (UAM), which quantifies each agent's prediction difficulty, enabling better resource allocation in interaction modeling. UAM not only enhances interaction modeling in Gameformer but also improves prediction accuracy and efficiency.

B. Learning-Based Path Planning

Traditional rule-based planning algorithms typically consist of steps like global path planning and local path planning, often using sampling, search, or optimization techniques. Learning-based planning algorithms (also known as end-to-end planning algorithms) directly map sensory inputs to trajectory or control signals. These methods mainly include imitation learning [9][10] and reinforcement learning [11][12]. Imitation learning constructs a dataset using expert trajectories and learns the mapping from this dataset. PlanTF [13] improves the encoding of kinematic states by introducing random dropout to force the encoder to learn the fundamental relationships between actions, greatly enhancing closed-loop simulation performance. PLUTO [14] is a framework based on imitation learning, data augmentation, and contrastive learning, which outperforms rule-based planners on the nuPlan dataset.

Given that both rule-based methods and learning-based approaches each have their respective strengths and weaknesses, we propose a Mixture of Experts model (MoE) that effectively integrates the advantages of both paradigms. By generating proposals from different methods, the model leverages a scoring module to select the final output, combining the benefits of each approach to improve overall planning performance.

III. Method

A. Definition and Design of UAM

Multimodal trajectory prediction involves utilizing the historical trajectories of agents and map information to predict their trajectories over a future period. Let X be historical trajectories of agents, $X = \{e^{-t_h}, e^{-t_h+1}, e^{-t_h+2}, ..., e^0\}$ where t_h denotes the duration of the observed historical trajectories, and $e^t(t \in -t_h, -t_h+1, ..., 0)$ includes attributes such as position coordinates, dimensions (length, width, height), orientation, velocity, and other relevant information at the given time step. Multimodal trajectory prediction requires generating K possible future trajectories for each agent, denoted by Y, over the prediction horizon t_f, along with predicting the probability P associated with each trajectory.

$$Y = \{s_1, s_2, ...s_K\} \tag{1}$$

$$P = \{p_1, p_2, ..., p_K\}, \sum_{i=1}^{K} p_i = 1 \tag{2}$$

s_i denotes the i-th trajectory and p_i is the corresponding probability score.

For each agent, its prediction uncertainty can be evaluated based on the K predicted trajectories and their corresponding probability scores. Firstly, we use L2 norm to measure the distance between different predicted trajectories. Let d_{ij} denote the average L2 distance between the i-th and the j-th predicted trajectories.Thus, an indicator s representing trajectory uncertainty can be constructed as follows:

$$s = \sum_{i=1}^{K} \sum_{j=1, j\neq i}^{K} p_i p_j d_{ij} \tag{3}$$

This formulation integrates the spatial diversity of trajectories d_{ij} with their likelihoods p_i and p_j, effectively capturing two key aspects of uncertainty. When the trajectories are closely clustered, d_{ij} values are smaller, leading to a lower

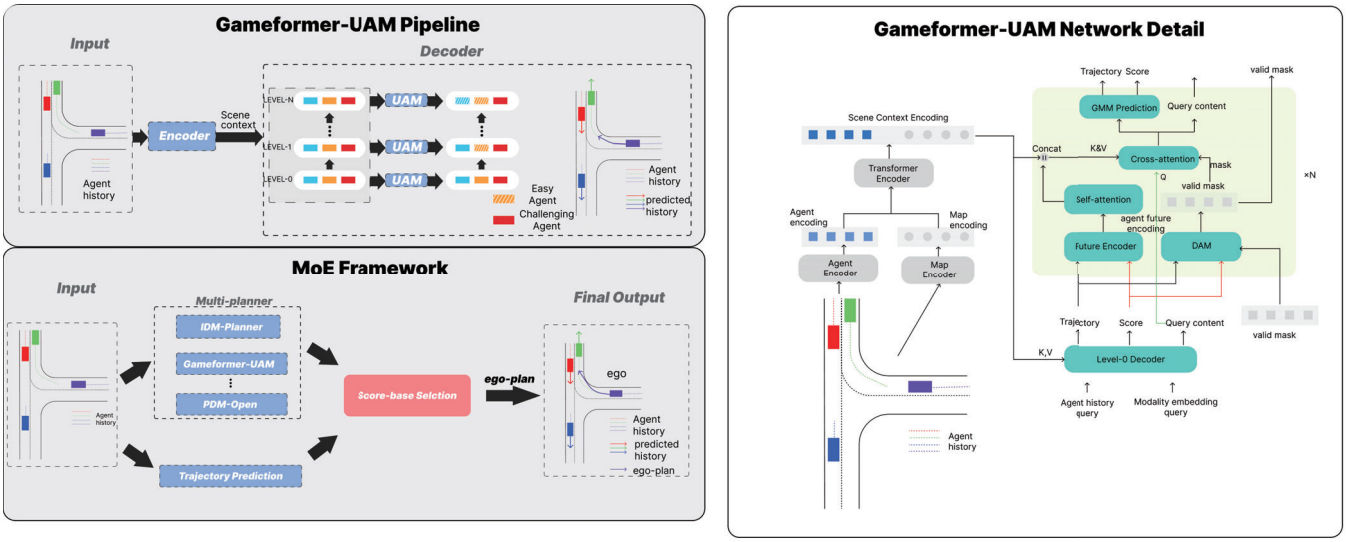

Fig. 1: Overview of our method. The Gameformer-UAM pipeline demonstrates the integration of the UAM module into the Gameformer framework, with the overall structure following an encoder-decoder design.The Gameformer-UAM Network Details highlight the specific network structure and MoE framework integrates both learning-based and rule-based approaches for planning.

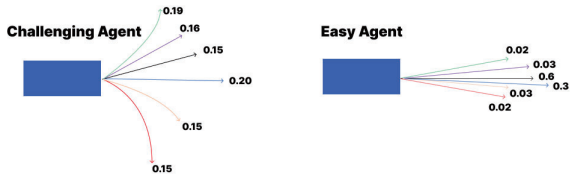

Fig. 2: Comparison of trajectory prediction distributions for challenging agents and easy agents

s, which indicates low uncertainty. Conversely, when the trajectories are more dispersed and the probabilities are evenly distributed, s increases, signifying higher uncertainty. Therefore, s serves as a comprehensive measure to evaluate the prediction uncertainty for an agent. Fig2 shows two examples of a challenging agent and an easy agent.

We give a theoretical proof when K=2 for this formula, the indictor can be simplified as:

$$s = p_1 p_2 d_{12} + p_2 p_1 d_{21} = 2 p_1 p_2 d_{12} = 2 p_1 (1 - p_1) d_{12} \quad (4)$$

When p_1 is fixed, the term d_{12} represents the spatial diversity of the two predicted trajectories. If the trajectories are close (low d_{12}), the uncertainty is low, and s becomes small. If they are far apart, s increases proportionally to the distance. When d_{12} is fixed, s is a quadratic function of p_1. s achieves its maximum value at $p_1 = 0.5$, which indicates that the probabilities of the two trajectories are equally likely and there is no clear preference for one trajectory over the other. As $p_1 \to 0$ or $p_1 \to 1$, s is close to 0 because one trajectory is overwhelmingly dominant (or certain), and the other is negligible. When K is larger than 2, this conclusion still holds.

Considering that different speeds may lead to varying distance differences d_{ij}, normalization is necessary for s. We introduce a normalized indicator s_{norm} to eliminate the influence of speed variations and ensure consistency across different scenarios.

$$s_{norm} = \frac{s}{\sum_{i=1}^{K} p_i l_i} = \frac{\sum_{i=1}^{K} \sum_{j=1, j \neq i}^{K} p_i p_j d_{ij}}{\sum_{i=1}^{K} p_i l_i} \quad (5)$$

where l_i represents the length of the i-th trajectory. This normalization ensures that the uncertainty indicator s is adjusted to account for differences in trajectory lengths, making the measure more robust to variations in speed.

B. Integration of UAM into Gameformer

Gameformer introduces a novel approach to enhance autonomous vehicle decision-making in complex environments by integrating game-theoretic principles with Transformer architectures, following the encoder-decoder framework.The encoder effectively captures relationships between scene elements to obtain scene context. The hierarchical decoder iteratively refines predictions by considering the outputs from previous levels, enabling nuanced modeling of agent interactions. The decoder consists of N layers. At each decoding layer, the model utilizes prior predictions and shared environmental context to progressively enhance the accuracy of interaction predictions. However, in each layer of the decoder, each agent updates its trajectory based on the predicted trajectories of other agents from the previous layer, without accounting for the varying prediction uncertainties of different agents.

By integrating the Uncertainty-Aware Module (UAM) to the decoder layers of Gameformer, we calculate the prediction uncertainty score for each agent at every decoding layer. This score is then compared against a predefined threshold.

If the score of an agent is below the threshold, the agent is excluded from subsequent updates, and its position is masked in the attention calculations. Conversely, if the uncertainty exceeds the threshold, the agent undergoes regular updates within the decoder. We update predictions and confidence using masked attention. For implementation details, please refer to Algorithm1.

Algorithm 1: Integration of UAM into Gameformer Decoder with Confidence Scores

Input: Decoder layers N, Agents' prior trajectory predictions P^{prev}, Confidence scores S^{prev}, Environmental context C, Threshold τ

Output: Updated agents' predictions P and confidence scores S

for $l \leftarrow 1$ **to** N **do**

  foreach *agent* i **do**

    $s_i \leftarrow \text{UAM}(P_i^{\text{prev}}, S_i^{\text{prev}}, C)$;

    // Calculate prediction uncertainty score for agent i

    if $s_i < \tau$ **then**

      $\text{mask}_i \leftarrow 1$; // Mask agent i in attention calculations

    else

      $\text{mask}_i \leftarrow 0$; // Allow regular update for agent i

  $P^{\text{new}}, S^{\text{new}} \leftarrow \text{DecoderLayer}(P^{\text{prev}}, S^{\text{prev}}, C, \text{mask})$; // Update predictions and confidence using masked attention

  $P^{\text{prev}} \leftarrow P^{\text{new}}$; // Pass updated predictions to the next layer

  $S^{\text{prev}} \leftarrow S^{\text{new}}$; // Pass updated confidence scores to the next layer

return P, S

C. Extention of UAM to MoE Planning Framework

Contemporary planning methods are typically divided into two main categories: rule-based and learning-based approaches. Rule-based methods provide high interpretability and reliability, but they often face challenges in adapting to dynamic environments and scaling effectively. Conversely, learning-based methods excel in adaptability and scalability but sometimes fall short in reliability compared to rule-based methods.

To leverage the strengths of both approaches and mitigate their respective limitations, we propose a unified prediction and planning framework built on the Mixture of Experts (MoE) paradigm. This framework generates diverse candidate trajectories using multiple planning methods and selects the optimal trajectory via a score-based evaluation mechanism.

The candidate trajectory generation process incorporates:

- **Rule-based methods**: such as IDM [15] , which offer robustness and interpretability.
- **Learing-based methods**: such as PDM-Open [16] and Gameformer-UAM, which combines advanced predictive capabilities with adaptability.

By integrating these methods, the framework achieves enhanced robustness and versatility in addressing diverse autonomous driving scenarios.

As illustrated in Fig. 1, the MoE framework comprises three main components:

- **Multi-Planner Module**: This module employs various planners, including IDM, PDM-Open with varying parameters, and Gameformer-UAM, to generate a wide range of candidate trajectories.
- **Trajectory Prediction Module**: It utilizes Gameformer-UAM to predict dynamic environments and refine trajectory candidates.
- **Score-Based Selection Module**: This component applies scoring standards from nuPlan, which account for comprehensive metrics, to evaluate and rank the trajectories. The highest-scoring trajectory is selected as the final output, ensuring optimal performance.

This structured approach enables the framework to seamlessly integrate diverse methodologies, combining their strengths to achieve superior performance, adaptability, and robustness in complex and varied autonomous driving scenarios.

IV. Experiment

A. Experimental Settings

1) **Dataset** NuPlan is the world's first large-scale autonomous driving planning benchmark platform, which provides a comprehensive framework for training machine-learning-based planners. It features a lightweight closed-loop simulator and dedicated motion planning metrics. Therefore, we use NuPlan for both training and testing. We extracted 63,687 scenarios for training and validation our model. For evaluating planning performance, we conducted tests on custom subsets, as well as the Test14-hard and Test14-random, and compared the results with other methods. The custom subsets consists of three types of scenes: highway, roundabout, and intersection, with 2,000, 2,000, and 3,000 scenarios, respectively.

2) **Evaluation metric** For multimodal trajectory prediction, common metrics include minimum Average Displacement Error (mADE), minimum Final Displacement Error (mFDE), and Miss Rate. mADE measures the smallest ADE value among the K predicted trajectories, while mFDE represents the smallest FDE value. These metrics help evaluate the accuracy and reliability of trajectory predictions.

Planning tasks are evaluated using diverse metrics. Open-loop planning typically employs metrics like ADE and FDE, similar to those used in trajectory prediction. In contrast, closed-loop planning relies on metrics such as Success Rate, Collision-Free Rate, and On-Route Rate. On-Route Rate measures the proportion of the route where the vehicle remains on the road, while Collision-Free Rate represents the percentage of scenarios without collisions. Success Rate

assesses whether the ego vehicle's progress along the expert trajectory exceeds a predefined threshold, such as 0.85. In NuPlan, scenario scores are derived by comparing expert and ego-driven trajectories, with higher scores reflecting better planning performance.

3) **Implementation Details** We conduct experiments on an NVIDIA RTX-4090 GPU. When training Gameformer with the Uncertainty-Aware Module (UAM), the threshold for UAM is set to 0.06. And the Gameformer model consists of a 3-layer encoder and a 3-layer decoder. The model is trained for 100 epochs with an initial learning rate of 0.001, following a MultiStepLR schedule where the learning rate is halved at epochs 10, 12, 14, 16, and 18. The batch size is set to 128. The training process utilizes 63,687 randomly sampled scenarios, each containing 1 second of historical data and the corresponding 8 seconds of future prediction trajectories. The dataset is split into a training set and a validation set in a 4:1 ratio, resulting in 51,069 scenarios for training and 12,768 scenarios for validation.

B. Trajectory Prediction Experiments

To validate the effectiveness of the Uncertainty-Aware Module (UAM), we compared the performance of Gameformer with and without UAM in trajectory prediction. As Gameformer and Gameformer with UAM provide predicted trajectories for both the ego vehicle and other agents, we utilize PlannerADE and PlannerFDE to evaluate the prediction accuracy of the ego vehicle, and PredictorADE and PredictorFDE for the accuracy of other agents' trajectories. Table I presents the performance metrics for an 8-second prediction horizon. The results show improvement across all metrics when using UAM, demonstrating that modeling prediction uncertainty awareness enhances trajectory prediction performance.

TABLE I: Comparison of Gameformer and Gameformer with UAM for an 8-second prediction horizon. Lower values indicate better performance.

Method	Pl-ADE ↓	Pl-FDE ↓	Pr-ADE ↓	Pr-FDE ↓
Gameformer	0.9356	1.9324	1.1127	2.1717
Gameformer-UAM	0.8653	1.6493	1.0447	2.0773

We compared the FLOPs of Gameformer and Gameformer-UAM, as shown in Table II. The result demonstrates that incorporating UAM into Gameformer reduces the overall computational cost.

TABLE II: Comparison of FLOPs per batch size between Gameformer and Gameformer-UAM.

Model	FLOPs (G) ↓
Gameformer	1.3512
Gameformer-UAM	**1.3480**

The Figure3 illustrates the masking proportions at different layers of Gameformer when the threshold is set to 0.06. It shows that the masking ratio increases with each successive layer, indicating that the predicted trajectories become progressively more deterministic as the layers deepen, with the entire process converging.

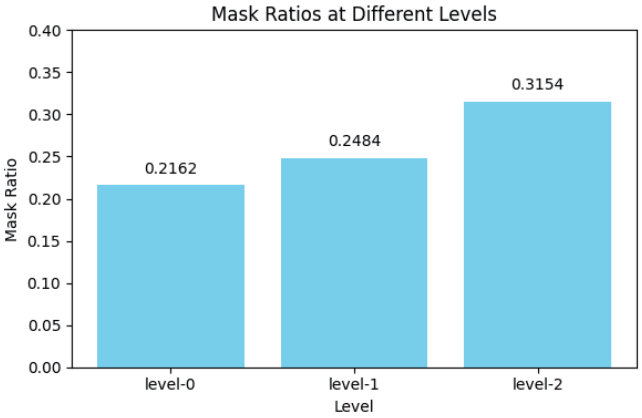

Fig. 3: Ratios of Masking at Different Decoder Layers with a Threshold of 0.06 in Gameformer

C. Planning Experiments

In the planning experiments, we primarily conduct tests on three datasets: custom dataset, Test14-hard [16], and Test14-random [16]. For the custom dataset, we conduct simulations using the most challenging reactive closed-loop approach. We compare the performance of our method across highway, roundabout, and intersection scenes against rule-based and learning-based methods on key metrics such as success rate, collision-free rate, and on-route rate. As shown in TableIV, our method consistently outperforms others across all evaluated metrics.

Additionally, we perform experiments on the Test14-hard and Test14-random datasets from the NuPlan Challenge and compare the results with several state-of-the-art methods. The evaluation includes tests on open-loop planning, non-reactive closed-loop planning, and reactive closed-loop planning. The results are summarized in TableIV. As shown in the table, our method achieves the best performance in both non-reactive and reactive closed-loop planning and also obtains competitive scores in open-loop evaluation. The high scores in closed-loop testing highlight the potential value of our approach in real-world applications.

V. CONCLUSIONS

This paper introduces the Uncertainty-Aware Module (UAM), a novel enhancement designed to improve trajectory prediction and ego-planning in autonomous driving. By evaluating and adapting to the varying prediction complexities of different agents, UAM significantly enhances the performance of the Gameformer model, achieving lower error metrics and reduced computational costs.

The integration of UAM within a Mixture of Experts (MoE) planning framework further demonstrates its versatility and effectiveness. Our method achieves state-of-the-art results on both custom and NuPlan datasets, surpassing

Method	Scene	Success Rate ↑ (%)	Collision-Free Rate ↑(%)	On-Route Rate ↑ (%)
IDM [15]	Intersection	54.10	88.63	88.43
	Roundabout	39.55	77.37	82.45
	Highway	72.29	91.67	93.05
PDM-closed [16]	Intersection	73.33	**98.30**	99.10
	Roundabout	62.60	93.40	98.85
	Highway	86.25	97.85	99.40
PDM-open [16]	Intersection	19.00	86.18	83.63
	Roundabout	22.60	60.70	73.60
	Highway	22.80	91.67	92.80
PlanTF [13]	Intersection	42.30	93.92	93.17
	Roundabout	41.70	89.20	98.30
	Highway	48.50	95.10	92.40
PDM-hybrid [16]	Intersection	73.40	98.25	99.07
	Roundabout	63.70	**95.35**	99.45
	Highway	86.50	**97.90**	99.45
Ours	Intersection	**80.53**	98.12	**99.40**
	Roundabout	**71.70**	94.80	**99.65**
	Highway	**88.90**	97.62	**99.50**

TABLE III: Comparison of success rate, collision-free rate, and on-route rate of different methods on various scenes.

TABLE IV: Performance comparison on the Test14-random and Test14-hard datasets. Performance metrics for other methods are taken from PlanTF.

Method	Test14-random			Test14-hard		
	OLS	NR-CLS	R-CLS	OLS	NR-CLS	R-CLS
IDM [15]	34.15	70.39	72.42	20.07	56.16	62.26
PDM-Closed [16]	46.32	90.05	91.64	26.43	65.07	75.18
GameFormer	79.35	80.80	79.31	75.27	66.59	68.83
PDM-Hybrid [16]	82.21	90.20	91.56	73.81	65.95	75.79
RasterModel [3]	62.93	69.66	67.54	52.4	49.47	52.16
UrbanDriver [11]	82.44	63.27	61.02	76.9	51.54	49.07
GC-PGP [17]	77.33	55.99	51.39	73.78	43.22	39.63
PDM-Open	84.14	52.80	57.23	79.06	33.51	35.83
PlanTF [13]	**87.07**	86.48	80.59	**83.32**	72.68	61.70
Ours	81.19	**90.79**	**91.57**	75.32	**71.01**	**76.96**

existing approaches in success rate, collision-free rate, and on-route rate.

In summary, UAM provides a strategic advantage for autonomous driving by optimizing resource allocation for prediction tasks. This work establishes a new benchmark for trajectory prediction and planning.

REFERENCES

[1] S. Shi, L. Jiang, D. Dai, and B. Schiele, "Motion transformer with global intention localization and local movement refinement," *Advances in Neural Information Processing Systems*, vol. 35, pp. 6531–6543, 2022.

[2] Z. Huang, H. Liu, and C. Lv, "Gameformer: Game-theoretic modeling and learning of transformer-based interactive prediction and planning for autonomous driving," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023, pp. 3903–3913.

[3] H. Caesar, J. Kabzan, K. S. Tan, W. K. Fong, E. Wolff, A. Lang, L. Fletcher, O. Beijbom, and S. Omari, "nuplan: A closed-loop ml-based planning benchmark for autonomous vehicles," *arXiv preprint arXiv:2106.11810*, 2021.

[4] J. Gao, C. Sun, H. Zhao, Y. Shen, D. Anguelov, C. Li, and C. Schmid, "Vectornet: Encoding hd maps and agent dynamics from vectorized representation," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 11 525–11 533.

[5] H. Zhao, J. Gao, T. Lan, C. Sun, B. Sapp, B. Varadarajan, Y. Shen, Y. Shen, Y. Chai, C. Schmid, *et al.*, "Tnt: Target-driven trajectory prediction," in *Conference on Robot Learning*. PMLR, 2021, pp. 895–904.

[6] J. Gu, C. Sun, and H. Zhao, "Densetnt: End-to-end trajectory prediction from dense goal sets," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 15 303–15 312.

[7] J. Ngiam, B. Caine, V. Vasudevan, Z. Zhang, H.-T. L. Chiang, J. Ling, R. Roelofs, A. Bewley, C. Liu, A. Venugopal, *et al.*, "Scene transformer: A unified architecture for predicting multiple agent trajectories," *arXiv preprint arXiv:2106.08417*, 2021.

[8] Z. Zhang, A. Liniger, C. Sakaridis, F. Yu, and L. V. Gool, "Real-time motion prediction via heterogeneous polyline transformer with relative pose encoding," *Advances in Neural Information Processing Systems*, vol. 36, 2024.

[9] S. Hu, L. Chen, P. Wu, H. Li, J. Yan, and D. Tao, "St-p3: End-to-end vision-based autonomous driving via spatial-temporal feature learning," in *European Conference on Computer Vision*. Springer, 2022, pp. 533–549.

[10] S. Teng, L. Chen, Y. Ai, Y. Zhou, Z. Xuanyuan, and X. Hu, "Hierarchical interpretable imitation learning for end-to-end autonomous driving," *IEEE Transactions on Intelligent Vehicles*, vol. 8, no. 1, pp. 673–683, 2022.

[11] O. Scheel, L. Bergamini, M. Wolczyk, B. Osiński, and P. Ondruska, "Urban driver: Learning to drive from real-world demonstrations using policy gradients," in *Conference on Robot Learning*. PMLR, 2022, pp. 718–728.

[12] J. Wu, Z. Huang, W. Huang, and C. Lv, "Prioritized experience-based reinforcement learning with human guidance for autonomous driving," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 35, no. 1, pp. 855–869, 2022.

[13] J. Cheng, Y. Chen, X. Mei, B. Yang, B. Li, and M. Liu, "Rethinking imitation-based planners for autonomous driving," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2024, pp. 14 123–14 130.

[14] J. Cheng, Y. Chen, and Q. Chen, "Pluto: Pushing the limit of imitation learning-based planning for autonomous driving," *arXiv preprint arXiv:2404.14327*, 2024.

[15] M. Treiber, A. Hennecke, and D. Helbing, "Congested traffic states in empirical observations and microscopic simulations," *Physical review E*, vol. 62, no. 2, p. 1805, 2000.

[16] D. Dauner, M. Hallgarten, A. Geiger, and K. Chitta, "Parting with misconceptions about learning-based vehicle motion planning," in *Conference on Robot Learning*. PMLR, 2023, pp. 1268–1281.

[17] M. Hallgarten, M. Stoll, and A. Zell, "From prediction to planning with goal conditioned lane graph traversals," in *2023 IEEE 26th International Conference on Intelligent Transportation Systems (ITSC)*. IEEE, 2023, pp. 951–958.