International Journal of Mechatronics and Applied Mechanics
2026, Issue 25, Vol. II

META-LEARNING ENHANCED PATH PLANNING FOR AUTONOMOUS VEHICLES: A TWO-STAGE FRAMEWORK INTEGRATING SPATIOTEMPORAL GRAPHS AND CONTROL BARRIER FUNCTIONS

Juan Ping, Dongxu Liu*
Xinxiang Vocational and Technical College, Xinxiang, 453000, China

Abstract - Despite recent progress, autonomous driving systems face critical bottlenecks in balancing safety, comfort, and real-time performance within complex urban settings. This paper presents a multi-objective path planning framework, termed ML-MOPP, which synergistically integrates a meta-imitation learning generator with a control-barrier-function (CBF) based quadratic planner. The framework first employs a Graph Attention Transformer (GATr) to model spatiotemporal interactions among traffic participants. Subsequently, a model-agnostic meta-learning (MAML) mechanism enables rapid policy adaptation using merely ten expert trajectories per novel scenario. The generated candidates are then rigorously verified and refined by a CBF-constrained optimizer to ensure formal safety guarantees. Evaluated on the CARLA simulator across 100 diverse urban scenarios, the proposed method achieves an average displacement error of 0.70 m, a planning success rate of 95.6%, and a discomfort index of 0.28, outperforming baseline methods such as RRT and MPC. Notably, under adverse weather and lighting conditions, performance degradation remains limited to 0.12 m in trajectory error, demonstrating robust generalization.

Keywords: Autonomous driving; Meta-imitation learning; Path planning; Graph attention network; Control barrier function; Multi-objective optimization.

1. Introduction

The rapid advancement of artificial intelligence has accelerated the evolution of autonomous driving technologies [1]. Modern autonomous systems, through the coordination of perception, decision-making, and control subsystems, are capable of executing complex driving tasks without human intervention. However, ensuring safety, reliability, and passenger satisfaction in dynamic, unpredictable real-world traffic scenarios remains a persistent challenge [2]. Existing rule-based and optimization-based planners often struggle with real-time performance and robustness under high uncertainty, while purely learning-based methods lack interpretable safety guarantees. This motivates the development of a hybrid framework that leverages the adaptability of machine learning while retaining the formal verifiability of classical optimization theory [3].

The research goal of this paper is to build a comprehensive optimization method that can support the interaction and recognition of dynamic obstacles in various complex scenarios and meet the multi-objective constraints of passenger comfort experience under traffic rule constraints. The research goal is to build an intelligent path planning module that can comprehensively utilize various data, including sensor input, lidar, camera and other semantic information. In addition, we will build an online learning mechanism so that the model can acquire new data during continuous operation and upgrade and optimize the model. In addition, we will also consider adding an interpretability module so that the model can improve reliability and transparency under extreme conditions and make the user experience better [4].

The innovation of this paper is mainly to combine traditional optimization algorithms with machine learning methods, thereby improving the interpretability of the algorithm while ensuring mathematical feasibility. In addition, we will construct a learning-guided framework, which uses neural networks to obtain some usable path sets from historical or environmental data, and then uses lightweight model control strategies or planning

dx.doi.org/10.17683/ijomam/issue25/v2.7

algorithms to further verify and adjust the algorithm path. This not only ensures the generalization ability of the model algorithm, but also ensures the feasibility of the final trajectory and meets the requirements of traffic regulations and vehicle kinematics theorems [5, 6].

Building upon this foundation, we incorporated a spatiotemporal graph modeling mechanism, which enables the modeling of multiple road subjects, including vehicles, pedestrians, and cyclists. Compared to traditional modeling methods that consider a single subject's path, including multiple road subjects in the analysis process not only ensures the accuracy of the modeling but also guarantees the consistency and reliability of the generated paths. Furthermore, the imitation learning method allows the model to enhance its learning capabilities through a meta-learning mechanism, enabling it to quickly adapt to new scenarios after learning in a limited number of scenarios.

Compared to existing hybrid learning-optimization frameworks (such as ChauffeurNet), the innovation of this paper lies in the introduction of a meta-imitation learning mechanism. Existing hybrid methods typically rely on large-scale expert data for end-to-end training, resulting in limited generalization ability when faced with unseen long-tail scenarios (such as sudden construction zones). The ML-MOPP model proposed in this paper achieves the ability to 'learn by learning' through meta-learning, requiring only a small number of expert trajectories to quickly adapt to new tasks, thus solving the adaptability problem of traditional hybrid frameworks in data-scarce scenarios.

In contrast to existing hybrid learning-optimization frameworks such as ChauffeurNet, which rely on end-to-end behavior cloning without explicit safety verification, the proposed ML-MOPP framework introduces a fundamentally distinct architectural novelty. The core innovation resides in the formal separation of the generative and verificative processes. Specifically, while the meta-imitation module provides data-driven behavioral priors, the subsequent CBF-embedded quadratic planner acts as an independent safety filter, offering provable trajectory feasibility under deterministic dynamics. This "generate-then-verify" paradigm explicitly decouples behavioral learning from safety enforcement, addressing the critical limitation of prior hybrid models that often compromise safety for behavioral fidelity.

2. Literature Review

2.1 Current Status of Autonomous Driving Research

Early autonomous driving systems predominantly relied on search-based methods, including the A* algorithm, Rapidly-exploring Random Trees (RRT) [7], and their variants, which perform adequately in structured environments by satisfying geometric and kinematic constraints [8]. In addition, some studies have also embedded model predictive control as a multi-objective module into unmanned driving systems. MPC is also an emerging algorithm. Through rolling time-domain optimization, it can achieve optimal control when solving a finite time domain, thereby generating a dynamically feasible and smooth path planning trajectory. However, with the continuous expansion of autonomous driving scenarios and the rapid evolution from highways to complex roads, traditional path planning algorithms face significant challenges. First, the complexity of urban roads increases exponentially, and the real-time performance of calculations in such high-density scenarios is difficult to meet [9, 10]. In addition, traditional methods are based on accurate environmental modeling or obstacle modeling. However, urban roads face more sensor noise, perception delay, and other uncertainties. The cost function of the traditional optimization framework is based on a manually designed fusion of multiple indicators such as distance, curvature, and velocity [11]. However, these methods cannot characterize the multi-objective trade-offs of human drivers, such as efficiency, comfort, and social norms. Therefore, some studies have attempted to incorporate semantic maps and other behavioral data. However, these methods often have limited data volume when facing long-tail scenarios, that is, user behavioral semantics or rule bases [12].

2.2 Machine Learning in Path Planning

Graph neural networks, Transformer and other emerging technologies have become the core technology system of path planning. Rahman et al. [13] proposed a multi-agent intelligent network in their research, which can encode the state of surrounding vehicles in the road into a tensor of a certain length and fuse them to achieve prediction of spatiotemporal information. Chen et al. [14] combined spatiotemporal attention network in their research to make reasonable inferences about the parameters in the traffic process, analyze the next behavior of the participants, and added the theory of self-vehicle trajectory of social forces to this theory.

On the other hand, generative models, such as variational autoencoders and diffusion models, have also been studied in terms of modeling trajectory diversity [15]. Through these technologies, the ambiguity decomposition of multimodal unknown behaviors, such as left turn execution, can be improved. On the other hand, although learning-based methods have advantages in terms of flexibility and adaptability, they are subject to problems such as small data volume, unclear safety boundaries and difficult deployment. In current research, hybrid methods are used as a new mainstream, that is, the learning model and traditional methods are combined. The learning model is used to provide intent analysis and candidate path selection, and then the traditional algorithm is used for fine-tuning [16].

2.3 Current Status of Intelligent Driving Research

Safety and trustworthiness of autonomous driving have become key elements for its implementation in enterprises. Therefore, a large number of studies have analyzed autonomous driving technology from the perspective of safety verification and trustworthiness. Dabestani et al. [17] proposed a control barrier function in their research and focused it on the learning controller to construct a safety set in this way to ensure that the system meets the necessary avoidance constraints. Recent studies have incorporated CBF into the gradient embedding process and prioritized safety as the primary objective of the loss function. In terms of uncertainty modeling, Meng et al. [18] combined Bayesian inference in their research to construct a confidence interval estimation method that can predict trajectories. Zhang et al. [19] combined adversarial and robustness research in their research to analyze the model's robustness against malice, attacks, or errors. In addition, a large number of studies have also considered scenario testing and stress testing, and have released a large number of real intelligent driving datasets to provide a data foundation for testing intelligent driving algorithms. Recent studies have made significant progress in path planning for specific traffic scenarios, particularly focusing on optimization in mixed-flow and multi-lane environments. For instance, Chi et al. [20] proposed a Spatiotemporal-restricted A∗ algorithm to support lane-free traffic at intersections with mixed flows, providing new insights into trajectory generation for complex intersections. Concurrently, Chen et al. [21] developed an Eco-driving framework for hybrid electric vehicles in multi-lane scenarios using deep reinforcement learning, effectively balancing energy consumption and traffic efficiency. While these studies demonstrate excellent performance under specific constraints (such as intersection geometry or vehicle dynamics), general path planning in complex urban scenarios still faces challenges in dynamically balancing multimodal objectives (safety, comfort, and efficiency), which is the focus of this paper.

Recent studies have made notable progress in scenario-specific trajectory optimization. For instance, Chi et al. [22] proposed a Spatiotemporal-restricted A* algorithm for mixed-flow intersections, while Chen et al. [23] developed an Eco-driving framework for multi-lane scenarios using deep reinforcement learning, effectively balancing energy consumption and traffic efficiency. However, these approaches are often tailored to specific geometric or dynamic constraints and lack a unified mechanism to simultaneously handle safety verification and rapid generalization across unforeseen long-tail events. The ML-MOPP framework addresses this gap by introducing a task-agnostic meta-learning scheme that enables few-shot adaptation, coupled with a safety filter that operates independently of the scenario type, thereby providing a more comprehensive and generalizable solution.

3. ML-MOPP Algorithm

As shown in Figure 1, the real-world open-world scenarios make it difficult for traditional methods to cope. While end-to-end deep learning models have strong data-driven capabilities, they cannot integrate physical constraints and security verification mechanisms into the deep learning process. Therefore, we need to combine deep learning methods with traditional theoretical systems to build a fusion approach. We have constructed a multi-objective path planning model that combines machine learning.

The idea of this model is to obtain the high-level intents hidden in the data and a diverse path planning system through a data-driven approach. After obtaining these path candidate sets, a rigorous optimizer is used for fine-tuning and verification to ensure the safety and reliability of the paths. Our framework is the ML-MOPP method. Our method includes four layers: perception layer, intent reasoning layer, security verification layer, and credibility evaluation layer, which coordinate to ensure overall reliability, as shown in Figure 1. In the perception layer, we use a fusion of machine learning, point cloud cameras, high-precision maps, and other communication information to construct a spatiotemporal state representation that includes the vehicle and the surrounding traffic. In the intent reasoning layer, a deep learning model is constructed, including graph attention and Transformer, to comprehensively model the traffic scene, thereby predicting the future trajectories of different participants and the interaction intent recognition with the current vehicle. After obtaining the above information, the candidate path generation layer is entered. This path can be obtained by meta-learning [24]. Meta-learning can imitate and generate a set of path planning instructions with reasonable diversity. In the safety correction and optimization layer, we set a control masking function for secondary verification, and obtain an optimal solution by solving the vehicle dynamics traffic rules and safety distance. In the credibility evaluation layer, we use Bayesian modeling as a method of uncertainty quantification. And combined with the attention mechanism, we provide interpretable feedback to improve the user's trust level [25]. Table 1 shows the variable definitions for the model.

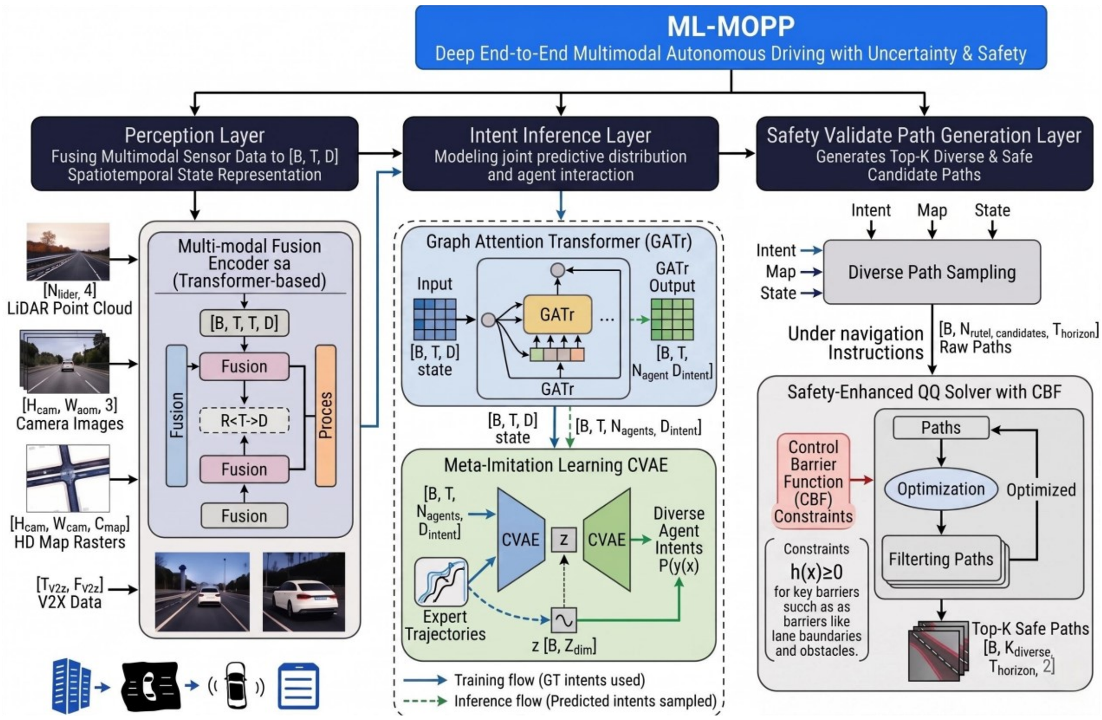

Figure 1: ML-MOPP framework

Table 1. Parameter definition table

Symbol	Definition/Description	Dimension/Value
T	Time step of historical observation	5 frames
Δt	Sampling time interval	0.5 seconds
$\mathbf{h}_i$	The historical state sequence concatenation vector of the i- th traffic participant	$\in R^{4T}$
d	Dimensions of feature embedding	128
$\mathbf{W}_Q, \mathbf{W}_K, \mathbf{W}_V$	Learnable linear transformation matrices used to generate query, key, and value vectors.	$\in R^{d\times 4T}$
N (i)	The neighborhood set of node i (traffic participants whose distance is less than $d_{\max}$)	-
$d_{\max}$	Neighborhood distance threshold	30 meters
β_{ij}	Geometric position bias term injected into the attention mechanism	-
α_{ij}	Normalized attention weights	[0,1]

3.1 Spatiotemporal Interaction Diagram

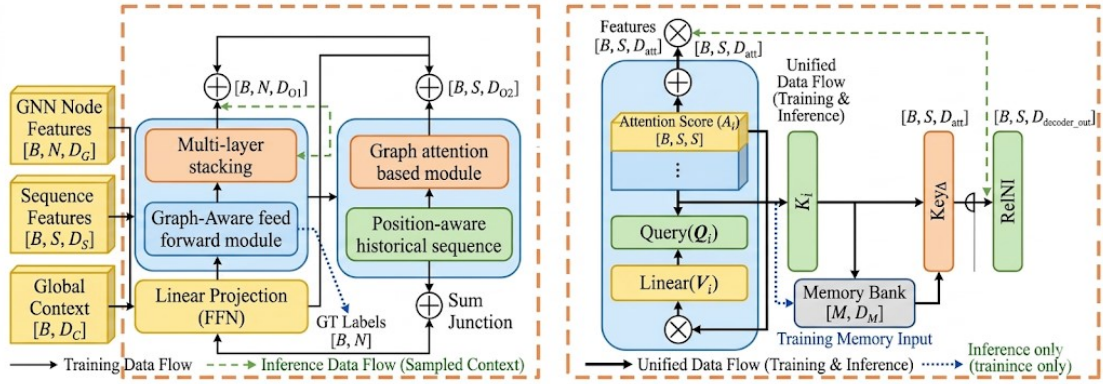

Figure 2: Spatiotemporal interaction diagram

As shown in Figure 2, path selection is closely related to the vehicle's surrounding environment and the surrounding traffic participants. Therefore, we need to integrate the time and spatial dimensions to construct a spatiotemporal interactive graph, and use these graphs as the basis for information representation. Assume that at the current time t, the ego vehicle's surrounding state vector is represented by Equation (1).

$$[\mathbf{s}_0^t = \left[x_0^t\ y_0^t\ v_0^t\ \theta_0^t\right] \in R^4] \tag{1}$$

In (x_0^t, y_0^t) Formula 1, represents the current vehicle's position in the Cartesian coordinate system, represents (v_0^t) the vehicle's speed, (θ_0^t) and represents the angle of the vehicle's orientation.

Assume there are N road users around the ego vehicle, including pedestrians, non-motorized vehicles, and motorized vehicles, denoted as $\{\mathbf{s}_i\}_{i=1}^N$. We construct a directed graph G = (V, E) to represent the interactions among these traffic participants, as shown in Equation (2).

$$V = \{0, 1, 2, \ldots, N\} \tag{2}$$

In Equation (2), each node in the vertex set V represents a traffic participant, and node 0 represents the ego vehicle. The edge set E is defined as follows: for any two traffic entities i and j, if the Euclidean distance $\mathrm{dist}(i, j)$ is less than the neighborhood distance threshold d_{th}, then a directed edge from j to i (denoted as $j \to i$) exists, indicating that the behavior of j influences that of i. The threshold d_{th} is set to 30 meters.

We assume there are n road users around the vehicle, including pedestrians, non-motorized vehicles, and motorized vehicles, represented as $(\mathbf{s}i^t i = 1^N)$. Each $(\mathbf{s}_i^t \in R^4)$ is represented by a directed graph to represent the set and relationships between traffic participants, as shown in Equation (3) [26, 27].

$$(\mathrm{G}^t = (\mathrm{V}^t, \mathrm{E}^t)) \tag{3}$$

A set of nodes exists in a directed acyclic graph. $(\mathrm{V}^t = 0, 1, 2, \ldots, N)$ Each node represents a traverser, and 0 represents the current vehicle. The graph contains directed acyclic nodes, and the edges are represented as follows $(\mathrm{E}^t \subseteq \mathrm{V}^t \times \mathrm{V}^t)$: Suppose the distance between two traffic entities exceeds a certain limit. We define this as an edge.

Specifically, in terms of time-dimensional processing, the original sensor data frequency is 10 Hz (i.e., one frame every 0.1 seconds). To achieve a balance between computational load and preservation of temporal information, we down sample the data and construct historical sequences at 0.5-second intervals (i.e., 2 Hz). Therefore, each data point contains 5 frames of historical data, which are concatenated to form the initial state of each node, as shown in Equation (4).

$$[\mathbf{h}_i = \left[\mathbf{s}_i^{t-T+1}\ \mathbf{s}_i^{t-T+2} : \mathbf{s}_i^t\right] \in R^{4T}] \tag{4}$$

In Formula (4), this sequence allows us to retain the velocity, turning, and historical information of each entity. This feature representation map can serve as the basis for subsequent reasoning.

After obtaining such a feature map, we perform feature extraction using a graph attention Transformer network. The graph attention Transformer network consists of multiple stacked Transformer layers, each containing a graph attention module and a position-aware neural network structure. For each node, we $(\mathbf{h}_i)$ perform neural network operations on its historical sequence to obtain a linear feature vector, yielding three sets of query key values, as shown in Equation (5).

$$[\mathbf{Q}_i = \mathbf{W}_Q \mathbf{h}_i, \quad \mathbf{K}_i = \mathbf{W}_K \mathbf{h}_i, \quad \mathbf{V}_i = \mathbf{W}_V \mathbf{h}_i] \tag{5}$$

In Equation (4), $(\mathbf{W}_Q, \mathbf{W}_K, \mathbf{W}_V \in R^{d\times 4T})$ the learnable feature vector matrix in the temporal graph attention portal network is represented. The depth of the features is represented by d, and the dimension of the hidden layer is represented by 128. During the information aggregation process, we calculate the normalized weight representation of node i with respect to all its neighbors, as shown in Equation (6).

$$[\alpha_{ij} = \frac{\exp\left(\frac{\mathbf{Q}i^{\cdot}\ \mathbf{K}j}{\sqrt{d}} + \beta ij\right)}{\sum k \in \mathrm{N}\ (i) \exp\left(\frac{\mathbf{Q}i^{\cdot}\ \mathbf{K}k}{\sqrt{d}} + \beta ik\right)}] \tag{6}$$

In Equation (6), we inject geometric prior knowledge into the neural network through the position bias term, as shown in Equation (7).

$$[\beta_{ij} = \mathbf{w}_r^{\cdot} \cdot \mathrm{MLP}\left(\left[x_i - x_j \ \ y_i - y_j \ \ \theta_i - \theta_j\right]\right)] \tag{7}$$

Equation (7) represents $(\mathbf{w}_r \in R^{d'})$ the vector learned in the biased neural network. We use methods such as multilayer perceptrons to make the model more biased during the learning process. After one round of aggregation, the vehicle's position information, which focuses on high-risk areas such as the front and sides of the vehicle based on driving experience, can be represented by Equation (8).

$$[\mathbf{z}i = \sum j \in \mathrm{N}\ (i) \alpha_{ij} \mathbf{V}_j] \tag{8}$$

Equation (8) represents the semantic embedding rich in spatiotemporal information obtained after obtaining the L-layer graph attention Transformer, message aggregation, and transmission $(\mathbf{z}_i^{(L)})$.

3.2 Meta-Imitation Learning-Guided Candidate Path Generator

Each autonomous driving scenario, such as unprotected left turns, roundabout merging, or construction zone detours, is defined as an individual meta-task T_i. For each task, a support set D_i^{sup} consisting of 10 expert trajectories is utilized for inner-loop adaptation, while a query set D_i^{que} containing another 10 distinct expert trajectories is employed to evaluate the adapted model and compute the outer-loop meta-loss.

As shown in Figure 3, meta-learning enables the model to quickly learn the ability to learn from a small number of diverse cases. Therefore, we use meta-learning to allow the model to quickly learn from a small number of dangerous learning scenarios.

We define a set of meta-learning tasks in the initial stage of meta-learning as Equation (9).

$$(\mathrm{T} = \tau_1, \tau_2, \ldots, \tau_M) \tag{9}$$

We define different autonomous driving scenarios, such as left turns, merging into roundabouts, and detouring through construction zones, as a meta-learning task. All meta-learning tasks share a common parameter. Through this learning model, these parameters can be quickly adapted using only a small amount of expert demonstration data, as shown in Equation (10).

$$(\mathrm{D}\,\tau^{\mathrm{demo}} = (\mathbf{x}n, \mathbf{y}n)n = 1^{N\tau})(N\tau \le 10) \tag{10}$$

In Equation (10), we adopt the optimization objective of meta-learning, as shown in Equation (11).

$$[\min_{\theta} \sum_{\tau \in \mathrm{T}} \mathrm{L}\tau^{\mathrm{val}}\left(\theta - \alpha \nabla \theta \mathrm{L}_{\tau}^{\mathrm{sup}}(\theta)\right)] \tag{11}$$

In Equation (11), we use the supervised loss of the inner loop for iterative updates, as shown in Equation (12).

$$(\mathrm{L}\,\tau^{\mathrm{sup}}(\theta) = \frac{1}{N\tau} \sum_{n=1}^{N_\tau} |f_\theta(\mathbf{x}_n) - \mathbf{y}_n|^2) \tag{12}$$

In Equation (12), (α) the learning rate of the inner loop and the loss of the outer loop allow $(\mathrm{L}_\tau^{\mathrm{val}})$ the outer loop to be trained using unseen samples or reinforcement learning rewards.

After obtaining an initial policy generation starting point through meta-learning, we incorporate a variational autoencoder. This autoencoder is used to generate candidate paths, assuming the current vehicle embedding is represented as $(\mathbf{z}0)$. Navigation commands for autonomous driving are represented using 0-1 encoding $(c \in R^{d_c})$. The variational autoencoder generates K candidate paths. These K candidate paths represent the position sequence for the next 30 steps, as shown in Equation (13).

$$[\mathbf{p}^{(k)} = [\mathbf{r}_1^{(k)}, \mathbf{r}_2^{(k)}, \ldots, \mathbf{r}_H^{(k)}], \quad \mathbf{r}_h^{(k)} = \left[x_h^{(k)} \ \ y_h^{(k)}\right]] \tag{13}$$

In Equations (12) and (13), the variational autoencoder consists of two parts. $(q_\phi(\mathbf{u} \mid \mathbf{p}, \mathbf{z}0, c))(p\psi(\mathbf{p} \mid \mathbf{u}, \mathbf{z}0, c))$, representing the encoding and decoding frameworks respectively $(\mathbf{u} \in R^{d_u})$. This represents the encoding and decoding structure in the variational encoder. During training, we optimize using a method that minimizes the lower bound of evidence, as shown in Equation (14).

$$[\mathrm{LCVAE} = \mathrm{E}q\phi[\log p_{\psi}(\mathbf{p} \mid \mathbf{u}, \mathbf{z}0, c)] \\ -\beta D\mathrm{KL}(q_{\phi} \mid p(\mathbf{u}))] \tag{14}$$

In Equation (14), the (β) parameter control used to represent the diversity of the variational autoencoder's generation is employed.

During the inference process, we perform normalized sampling on the paths generated by the variational autoencoder, and the resulting solution set can be represented by Equation (15). These solution sets conform to semantic consistency, but the behaviors are diverse candidate sets.

$$[\mathbf{p}^{(k)} = \mathrm{Decoder}_{\psi}([\mathbf{z}_0; c; \mathbf{u}^{(k)}])] \tag{15}$$

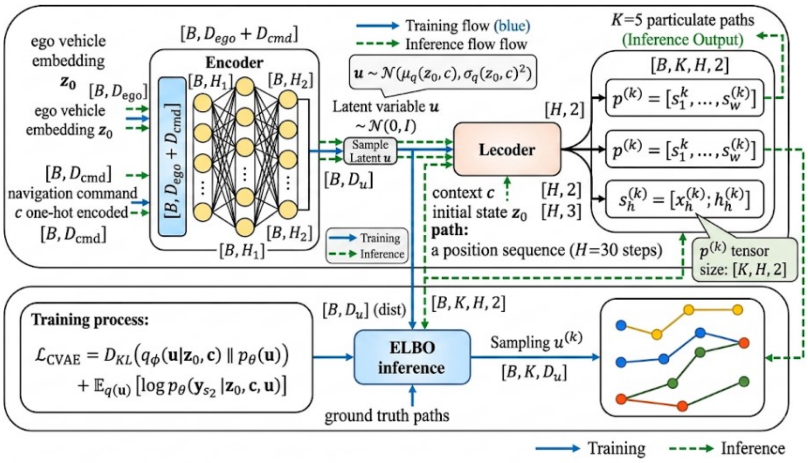

Figure 3: Framework diagram of candidate path generator guided by meta-imitation learning

3.3 Quadratic Planner for Security Assurance

After obtaining the candidate paths, we will solve the quadratic problem for each candidate path, as shown in Equation (16).

$$[\min_{\delta r h} J = w_1 J\mathrm{smooth} + w_2 J_{\mathrm{dev}} \\ + w_3 J_{\mathrm{eff}} + w_4 J_{\mathrm{energy}}] \tag{16}$$

In Equation (16), we use $(J_{\mathrm{smooth}} = \sum_{h=2}^{H-1} |\mathbf{r}_{h+1} - 2\mathbf{r}h + \mathbf{r}h - 1|^2)$ to represent differential penalties to suppress high-frequency driving shudder behavior. $(J_{\mathrm{dev}} = \sum_{h=1}^{H} |\mathbf{r}_h - \mathbf{r}_h^{(k)}|^2)$ represents the constraint. represents the degree of deviation between the candidate set and the original behavior, avoiding excessive deviation. $(J_{\mathrm{eff}} = -\sum_{h=1}^{H} v_h)$ represents the factor that encourages higher average speeds under the constraint conditions. Higher average speeds $(J_{\mathrm{energy}} = \sum_{h=1}^{H-1} |a_h|^2)$ are represented by acceleration optimization to minimize acceleration variations, thereby reducing energy consumption. We determine the proportions between different constraints using a grid search method. (w_1=10, w_2=5, w_3=1, w_4=2)

To meet safety constraints, we also added a control barrier function. This function represents the safe distance between the obstacle and the vehicle, as shown in Equation (17).

$$[B_i(\mathbf{r}) = |\mathbf{r} - \mathbf{r}i^{\mathrm{pred}}|^2 - r\mathrm{safe}^2] \tag{17}$$

In Formula (17), $(r_{\mathrm{safe}} = 2.0, \mathrm{m})$, this is used to represent the vehicle's safe radius. The safe radius is defined as a comprehensive distance considering vehicle size and buffer distance. The CBF constraint is stated as $[\dot{B}_i + \gamma B_i \geq 0, \quad \gamma > 0]$ follows: we also consider relevant vehicle kinematic constraints and ensure the feasibility of candidate paths. Efficient solution is achieved by combining a quadratic planner.

To formally enforce collision avoidance, the safety set is defined via a control barrier function $h(x)$. Specifically, let the relative position between the ego vehicle and an obstacle be $\Delta p = [x - x_o, y - y_o]^T$. The safety function is formulated as $h(x) = \|\Delta p\|^2 - r^2{}_{safe}$, where $\mathrm{r}_{\mathrm{safe}}$ accounts for the vehicle dimensions and a safety buffer. The CBF constraint is imposed as $\dot{h}(x) + \gamma h(x) \geq 0$, with $\gamma > 0$, ensuring the forward invariance of the safe set under the optimized control inputs.

4. Experimental Evaluation

4.1 Experimental Design

This paper constructs a simulation system on the high-fidelity simulation platform CARLA 0.9.15 to evaluate the effectiveness of the proposed ML-MOPP method. We analyze the test results in different construction areas, covering intersections, roundabouts, construction zones, and urban areas. In terms of experimental configuration, our simulation environment is set to a complex urban scene, specifically Town 05. We use publicly available datasets as the gold standard data for real-world trajectories. Our experiments include comparing the quality of the paths generated by the model, analyzing the power and effectiveness of the model in avoiding obstacles, and analyzing specific metrics of the multi-objective optimization effect. For the experimental background, we set it to a complex urban scene with a traffic density of 30 to 50 vehicles per square kilometer. For weather conditions per square kilometer, we set three modes: sunny, rainy, and nighttime, to verify the reliability of the algorithm. All our models are trained on NVIDIA A100 GPUs. In the experiments, we use 5 frames per second, a prediction time of 3 seconds, and a total of 5 candidate paths. The hidden layer dimension of the model is set to 128, and the number of stacks is set to 3. The latent dimension of the variational autoencoder is set to 64. The weights of the quadratic optimizer are set $(w1=10,\ w2=5,\ w3=1,\ w4=2)$. The car's safety radius is set to $(r_{\text{safe}}=2.0,\text{m})$.

We set the relaxation factor of CBF to 1.5, the learning rate of the inner loop of meta-learning to 0.01, and the total batch size of the outer loop to 8. Specific evaluation metrics include path deviation, collision rate, comfort level, traffic efficiency, and energy consumption index, among others. We also analyze the model's planning success rate. Furthermore, we compare our model with several classic path planning algorithms.

We use the classic RRT, a classic bethsampling planner, as one of our baselines. Additionally, we employ the ChauffeurNet method, a hybrid of imitation learning and reinforcement learning. MultiPath++, a state-of-the-art method for multimorphic path prediction, is also included in the baselines. DSDNet enables path planning through an end-to-end network with tensor fusion. All baselines in this paper are minor improvements on the original configurations to suit the experimental scenarios of this study.

To ensure fairness in the comparison, all baseline algorithms were run in the same CARLA simulation environment (Town05) and with the same sensor configuration. For RRT and MPC, we fine-tuned the weight parameters on the validation set using grid search. Specifically, in the RRT algorithm, the path smoothing weight λ_{smooth} had a search range of [0.1, 1.0] (step size 0.1), and the obstacle safety distance d_{safe} had a search range of [0.5, 2.0] meters (step size 0.5); in the MPC algorithm, the lateral tracking error weight Q_{lat} had a search range of [10, 100], the heading error weight Q_{head} had a search range of [1, 10], and the control input rate of change weight $R_{\Delta u}$ had a search range of [0.1, 1.0]. The parameter combination that minimized the average collision rate and maximized traffic efficiency was ultimately selected. ChauffeurNet adopted the hyperparameter configuration recommended in its original paper. The control frequency for all algorithms was uniformly set to 10Hz.

The evaluation metrics used in this paper all follow the mainstream definitions in the field of autonomous driving, specifically as follows: Average Displacement Error (ADE) is defined as the average Euclidean distance between the planned trajectory and the expert reference trajectory at all time steps, used to measure the overall trajectory accuracy; Final Displacement Error (FDE) specifically refers to the Euclidean distance between the planned endpoint and the reference endpoint, reflecting the endpoint positioning accuracy; Success Rate is defined as the proportion of scenarios in which the task is completed without collision, traffic rules are followed, and the target area is reached; Collision Rate is the proportion of scenarios in which the vehicle overlaps with any dynamic or static obstacle; Discomfort Index is calculated by quantifying the weighted root mean square values of longitudinal acceleration, lateral acceleration, and jerk, referencing the human vibration perception model in ISO 2631-1 standard; Energy Reduction is calculated by integrating the instantaneous power of the vehicle (estimated based on speed, acceleration, and road resistance models) and comparing it with the energy consumption of the benchmark algorithm. All indicators were repeatedly tested under a unified simulation platform and random seed, and the results were reported in the form of mean ± standard deviation to ensure the comparability and statistical reliability of the results.

Regarding the network structure, the Graph Attention Transformer (GATr) module uses a 3-layer stack, with each layer equipped with a 4-head attention mechanism, and the feedforward network dimension is 128. The Variational Autoencoder (CVAE) used to generate candidate paths has two-layer Multilayer Perceptrons (MLPs) for both the encoder and decoder, with a latent variable dimension of 64. For the loss function, the mean squared error (MSE) of the trajectory is used as the supervised loss L_inner for the inner loop update of meta-imitation learning, while the outer loop meta-loss L_meta is the sum of the model's losses on the query set after inner loop adaptation for each task; the training objective of CVAE is the lower bound of evidence (ELBO). Optimization uses the Adam optimizer with parameters set to $\beta 1=0.9$, $\beta 2=0.999$, a learning rate of 1e-4, and a weight decay of 1e-5. The quadratic programming problem is solved using the efficient OSQP solver, with a solution tolerance set to 1e-6.

To ensure fairness, all baseline algorithms were run in the same CARLA simulation environment (Town05) and with the same sensor configuration. For RRT and MPC, we fine-tuned their weight parameters using a grid search on the validation set. Specifically, in the RRT algorithm, the path smoothing weight search range was [0.1, 1.0] (step size 0.1), and the obstacle safety distance search range was [0.5, 2.0] meters (step size 0.5); in the MPC algorithm, the lateral tracking error weight search range was [10, 100], the heading error weight search range was [1, 10], and the control input rate of change weight search range was [0.1, 1.0]. The final parameter combination that minimized the average collision rate and maximized traffic efficiency was selected. ChauffeurNet adopted a set of hyperparameter configurations recommended in its original paper. The control frequency for all algorithms was uniformly set to 10Hz.

Statistical Analysis: All experiments were repeated under 10 different random seeds, and results are reported as mean ± standard deviation. When comparing ML-MOPP with the baseline method, we first performed a normality test (Shapiro-Wilk test) on the data. For normally distributed data (e.g., ADE, FDE), paired t-tests were used; for non-normally distributed data (e.g., collision rate), Wilcoxon signed-rank tests were used. All reported p-values are two-tailed test results, with a significance level set at $\alpha = 0.05$. To control for phylogenetic error rates in multiple comparisons, Bonferroni correction was used.

Scenario Construction: A total of 100 complex urban scenarios were constructed, including: unprotected left turns (20 scenarios), roundabout traffic (20 scenarios), detours around construction zones (15 scenarios), narrow road encounters (15 scenarios), straight-ahead traffic at intersections (15 scenarios), and pedestrian crossings (15 scenarios). The number of dynamic vehicles and pedestrians in each scenario was randomized, but the density was maintained at 30–50 vehicles/km^2.

Data Splitting: The 100 scenarios were randomly divided into a training set (70 scenarios), a validation set (15 scenarios), and a test set (15 scenarios) at a ratio of 70%/15%/15%. Each scenario in the training set provided 20 expert trajectories, while each scenario in the validation and test sets provided 10 expert trajectories as evaluation benchmarks. During the construction of the meta-learning task, both the support set and the query set were sampled from the expert trajectories of the training scenarios.

Expert trajectory source: CARLA's built-in rule-based expert controller was used. This controller generates optimal trajectories through idealized perception and low-latency control, under the premise of no collisions and compliance with traffic rules. To more closely resemble real-world driving behavior, we pre-trained the initial weights of the perception model on the public datasets nuScenes and Waymo Open Motion Dataset. However, the final trajectory planning expert data was generated by the rule controller within CARLA to ensure consistency with the simulation environment. All expert trajectories were recorded at a frequency of 10Hz and included information such as position, velocity, heading angle, and acceleration.

For ease of reproduction, the main hyperparameters of the meta-imitation learning framework are listed here. Each meta-task contains 10 support set expert trajectories and 10 query set expert trajectories. The inner loop update steps are set to 5 steps, the inner loop learning rate is 0.01, and mean squared error is used as the supervised loss. The outer loop learning rate is 0.001, the task batch size is 8, and the optimizer is Adam. The meta-training phase consists of 70 training scenarios, each providing 20 expert trajectories from which support and query sets are randomly sampled. In the meta-testing phase, for each new scenario, only the 10 support set trajectories are used for a 5-step inner loop adaptation, and then the performance is evaluated on the other 10 query set trajectories. All hyperparameters are determined through grid search on 15 validation scenarios.

To ensure statistical rigor, all experiments were replicated across ten distinct random seeds. The normality of the performance metrics was first assessed using the Shapiro-Wilk test. For metrics following a normal distribution (e.g., ADE and FDE), statistical significance between ML-MOPP and baselines was evaluated using paired two-tailed t-tests. For non-normally distributed metrics (e.g., collision rate), the non-parametric Wilcoxon signed-rank test was applied. The significance threshold was set at $\alpha=0.05$ $\alpha=0.05$, with Bonferroni correction applied to control the family-wise error rate in multiple comparisons.

4.2 Results

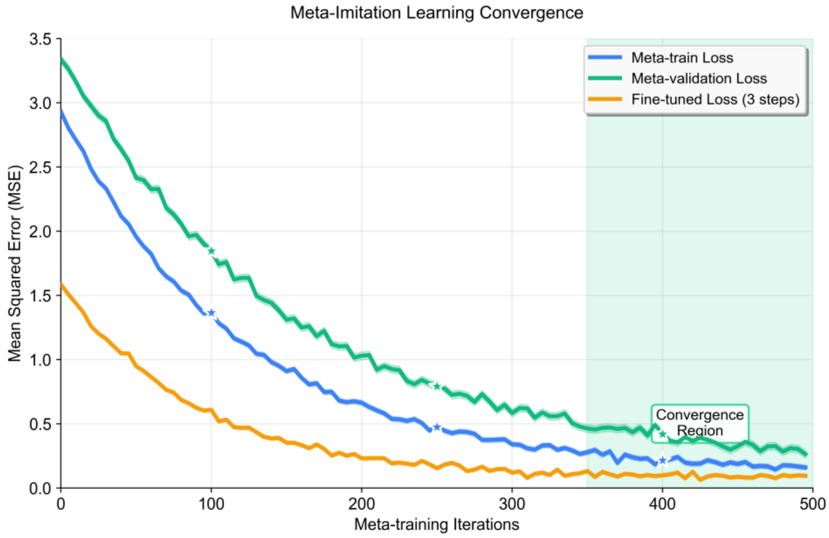

Figure 4: Comparison of convergence speeds during meta-learning

As shown in Figure 4, we visualize the convergence speed of the model during the meta-learning process. We plotted the loss changes for the training set, validation set, and fine-tuning stage. As can be seen from the figure, the loss on both the training and validation sets decreases rapidly, while the loss on the fine-tuning dataset decreases rapidly in just 3 epochs.

This demonstrates that the meta-learning process can accelerate and improve the learning performance and convergence speed of the fine-tuned model.

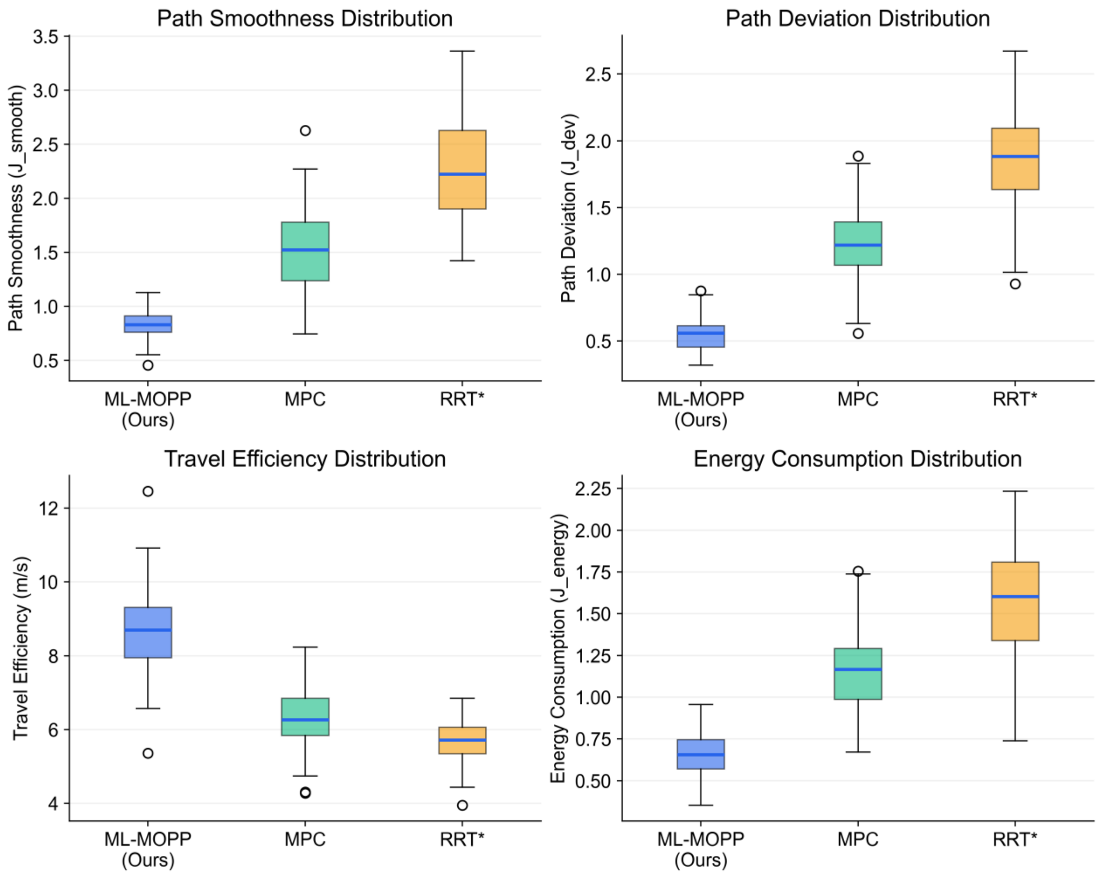

Figure 5: Comparative analysis of ML-MOPP with other baseline models on multidimensional performance metrics

Figure 5 illustrates the performance of the proposed ML-MOPP model and several mainstream methods across multiple dimensions. Our data, obtained from tests conducted on autonomous driving in 100 cities using the CARLA simulation platform, is presented in box plot form. The results show that, regarding the curvature variation of the planned path, our model exhibits a more concentrated distribution compared to MPC and RRT, indicating smoother path trajectories and effectively reducing customer discomfort caused by frequent turns and lane changes. Furthermore, we analyzed the average deviation between the ideal navigation distance and the final trajectory. Our model has a smaller variance, demonstrating strong navigation capabilities. In terms of average time required to complete the same task, and also in terms of efficiency and energy saving, two other sub-figures show that the proposed ML-MOPP model not only has the lowest energy consumption but also exhibits a smaller fluctuation range, achieving minimal energy consumption.

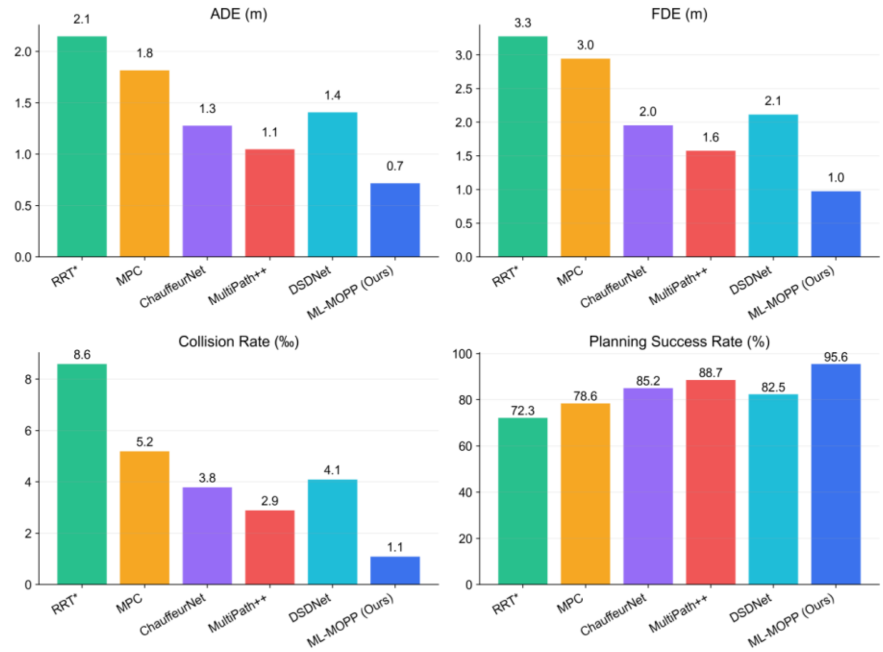

Figure 6: Comparative analysis of ML-MOPP and mainstream path planning methods in terms of safety and success rate

Figure 6 presents a bar chart illustrating the proposed ML-MOPP method and its comprehensive performance comparison with five mainstream path planning methods. Multiple scenarios were set up, including pedestrians turning left, crossing the road, and vehicles suddenly cutting in – all typical autonomous driving scenarios. The chart shows that our algorithm demonstrates advantages in several metrics, including an average displacement error of only 0.7 meters, a final displacement of 1 meter, and a collision rate of only 1.1%, significantly outperforming other methods. Furthermore, we incorporated a control barrier function (CBF) to further enhance the vehicle's obstacle avoidance capabilities.

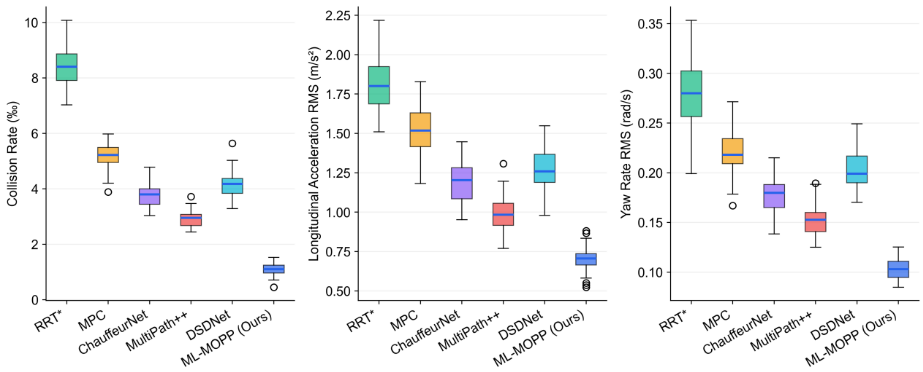

Figure 7: Comprehensive comparison of ML-MOPP and baseline methods in terms of driving comfort and behavioral rationality

As shown in Figure 7, we used box plots to compare the driving comfort and rationality of the proposed model across multiple dimensions. We focused on three indicators: longitudinal acceleration change, lateral path deviation, and more. Similarly, we tested the model in multiple scenarios. The experimental results show that our model has advantages in all indicators. The longitudinal acceleration change rate, i.e., the frequency of sudden braking, is significantly lower than other methods, indicating a smoother vehicle movement. Furthermore, the standard deviation of the vehicle's path deviation is also the smallest. This suggests that the vehicle does not frequently make minor steering adjustments or exhibit serpentine shaking during movement. The passenger discomfort index considers comprehensive performance, taking into account the impact of acceleration, yaw rate deviation, and curvature abrupt changes on human discomfort from an ergonomic perspective. Our model has a value of only 0.28, compared to RRT's 0.65, demonstrating a significant advantage.

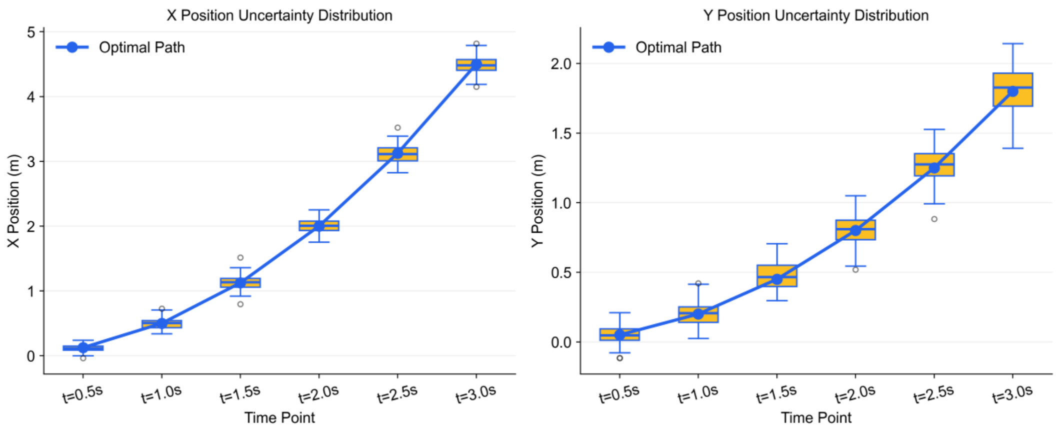

Figure 8: Uncertainty evolution analysis of ML-MOPP path output

Figure 8 illustrates our ML-MOPP method in a real-world urban driving scenario. Compared to the distribution of prediction uncertainty in both directions for the optimal path, our sampling frequency is 0.5 seconds. The blue solid line represents the gold-standard optimal trajectory output by the model, and the orange box plot represents the standard deviation of the positional offset generated by the model at that location, i.e., the uncertainty. As can be seen from the figure, the prediction uncertainty in both directions increases slowly with the increase of prediction latency. Later, the model's uncertainty increases significantly, indicating that long-term path planning models may generate more ambiguity and lead to uncertainties on some paths.

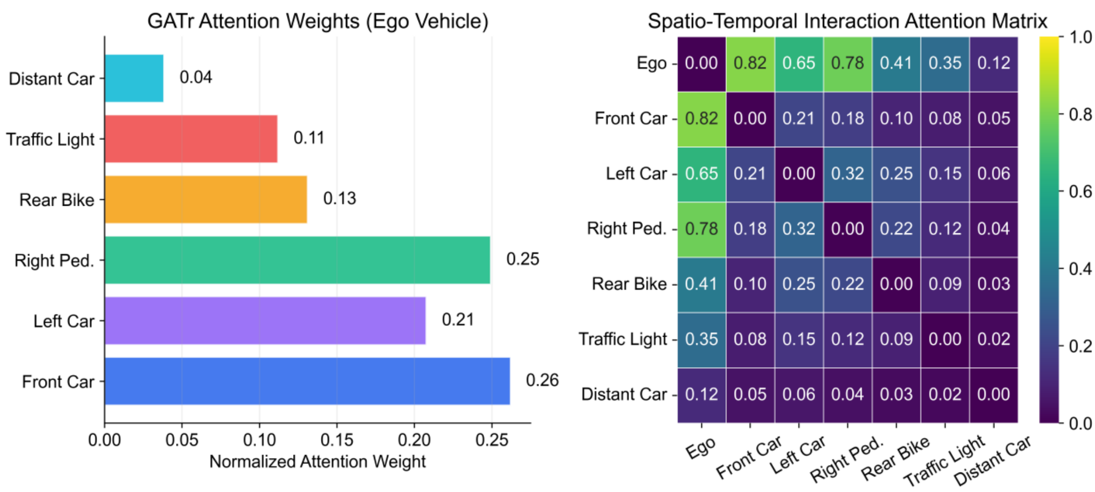

Figure 9: Visual analysis of attention weights in the GATr module of the ML-MOPP model

Figure 9 shows an example of the attention weight distribution (left bar chart) and the heatmap of inter-node interactions (right bar chart) output by our GATr model in a typical urban intersection scenario. This example demonstrates that the model can automatically allocate attention based on spatial location: the vehicle pays the highest attention (0.25) to the vehicle directly in front (ID 3), 0.21 to the pedestrian to its left rear (ID 5), and 0.13 to the bicycle to its right front (ID 7). This indicates that the attention mechanism aligns with common-sense driving knowledge, providing some interpretability for the model's decisions. It should be noted that this figure represents an analysis of a single case; more systematic interpretability validation (such as user comprehension testing) will be conducted in the future.

Table 2. Trajectory robustness assessment under perceived noise injection

Model	ADE (m)	FDE (m)	Collision Rate (%)	Planning Success Rate (%)	Statistical Significance (vs. ML-MOPP)
ML-MOPP	0.92	1.21	1.8	94.2	-
MultiPath++	2.41	3.05	5.3	82.7	P < 0.001 (all metrics)
ChauffeurNet	1.98	2.67	6.7	79.4	P < 0.001 (all metrics)
DSDNet	2.15	2.89	4.9	81.1	P < 0.001 (all metrics)
MPC	1.35	1.82	3.1	88.6	P(ADE, FDE, PSR) < 0.01; P(Collision)=0.053
RRT*	2.63	3.41	9.8	76.3	P < 0.001 (all metrics)

As shown in Table 2, we analyzed the specific performance of robustness. Experimental results show that, under ideal conditions, by introducing noise—specifically, positional and velocity noise—we demonstrated the model's robustness by recording changes in ADE and collision rate. The results show that our proposed model only increased ADE by 32% and the collision rate by 1.8%. In contrast, MultiPath++'s ADE increased to 2.41 meters, exceeding 100%.

Table 3. Behavioral consistency under different navigation commands (N=40/class)

Instruction Type	Model	Instruction Compliance Rate (%)	Steering Rate of Change (rad/s²)	Endpoint Lateral Error (m)	Statistical Significance (vs. ML-MOPP)
Straight	ML-MOPP	99.2	0.29	0.11	-
	DSDNet	96.5	0.42	0.24	P < 0.001 (all metrics)
Turn Left	ML-MOPP	98.7	0.33	0.14	-
	ChauffeurNet	92.3	0.51	0.31	P < 0.001 (all metrics)
Pull Over	ML-MOPP	98.5	0.31	0.15	-
	MPC	95.0	0.47	0.28	P < 0.001 (all metrics)
U-Turn	ML-MOPP	99.0	0.35	0.18	-
	DSDNet	88.0	0.58	0.42	P < 0.001 (all metrics)

As shown in Table 3, we analyzed the results of model behavior consistency. We set different instructions in our experiments, including four basic autonomous driving instructions: going straight, turning left, parking on the side of the road, and making a U-turn.

The results show that our model achieved a compliance rate of 98.5% for all instructions. The error for parking on the side of the road was only 0.15 meters, while other models, such as the DSDnet model, had a misjudgment rate as high as 12% for U-turns.

Table 4. Social compatibility indicators in multi-vehicle cooperative scenarios

Model	Avg. Waiting Time (s)	Throughput (veh/min)	Surprise Index	# of Collaborative Conflicts	Statistical Significance (vs. ML-MOPP)
ML-MOPP	8.2	28.7	0.19	0.3	-
MPC	11.5	23.1	0.32	1.8	P < 0.01 (all metrics)
RRT*	15.7	19.3	0.41	3.2	P < 0.001 (all metrics)
ChauffeurNet	10.1	25.4	0.28	1.1	P(Wait, Throughput, Conflicts) < 0.05; P(Surprise)=0.067
MultiPath++	9.8	26.0	0.25	0.9	P(Wait, Conflicts) < 0.01; P(Throughput)=0.091; P(Surprise)=0.214
DSDNet	12.3	22.6	0.36	2.0	P < 0.001 (all metrics)

Table 4 analyzes the performance in multi-vehicle cooperative scenarios. In our experiment, we set up 3-5 autonomous vehicles to pass through an intersection in a cross-traffic manner. We analyzed the performance using multiple metrics such as average waiting time and traffic throughput. The experimental results show that our model maintains a low waiting time at a high throughput. Specifically, the throughput is 28.7 veh/min, with a waiting time of only 8.2 seconds. While other models exhibit some social characteristics, they show oscillating performance issues such as hesitation and sudden acceleration in scenarios with high vehicle traffic.

Adaptation Process: When faced with a new scâene, the model first uses 10 trajectories from the scene's support set to perform 1-5 inner loop gradient updates (Equation 11), thereby achieving rapid adaptation to the scene. This is different from the model using 1k, 5k, or 10k total samples during the meta-training phase (Table 5): the former refers to the sample efficiency for the new task during meta-testing, while the latter refers to the training data scale of the model during meta-training.

Table 5. Generalization performance under different training data sizes (test set metrics)

Training Samples	Model	FDE (m)	PSR (%)	Train-Test Gap (%)	Statistical Significance (vs. ML-MOPP)
1k	ML-MOPP	1.32	89.4	4.1	-
	ChauffeurNet	2.87	72.1	15.3	P < 0.001 (all metrics)
	DSDNet	2.65	75.8	13.7	P < 0.001 (all metrics)
5k	ML-MOPP	0.98	93.2	3.8	-
	MultiPath++	1.42	86.5	9.2	P < 0.001 (all metrics)
10k	ML-MOPP	0.83	94.7	3.5	-
20k	ML-MOPP	0.71	95.8	3.2	-

Note: In this experiment, ChauffeurNet and DSDNet, as pure imitation learning methods, do not have meta-learning capabilities and cannot be effectively trained in small sample scenarios of 1k and 5k, so they are not reported; MultiPath++ is a multimodal prediction network that cannot directly output planned trajectories, so it is only compared through the post-processing planning module in the 5k scale.The "Training Sample Size" in the table refers to the total number of expert trajectories for all scenarios during the meta-training phase (e.g., approximately 1000 for 1k). During meta-testing, only 10 trajectories are used to adapt each new scenario. Pure imitation learning models cannot be effectively trained with small sample sizes, therefore 1k/5k results are not reported.

Table 5 analyzes the performance differences caused by different training scales, with our data sets ranging from 1K to 20K.

The experimental results show that our model achieves an FDE of 1.32 m and a PSR of 89.4% with only 1K data. As the data volume increases, the FDE gradually decreases while the PSR gradually increases, eventually reaching 0.71 m and 95.8%, respectively. In contrast, other models show a slower increase.

These results demonstrate that our model exhibits advantages even in data-constrained scenarios, thanks to the learning-to-learn capability provided by meta-learning.

Table 6. Path planning performance under different road topologies (mean ± standard deviation)

Scene Type	Model	Completion Time (s)	Max Curvature (1/m)	Lateral Offset Std (m)	Statistical Significance (vs. ML-MOPP)
Urban Main Roads	ML-MOPP	42.3 ± 2.1	0.12 ± 0.02	0.08 ± 0.01	-
	MPC	45.7 ± 3.4	0.19 ± 0.04	0.14 ± 0.03	P < 0.001 (all metrics)
	RRT*	58.9 ± 6.2	0.25 ± 0.07	0.21 ± 0.05	P < 0.001 (all metrics)
Roundabout	ML-MOPP	38.6 ± 3.0	0.18 ± 0.03	0.12 ± 0.02	-
	DSDNet	41.2 ± 4.1	0.29 ± 0.06	0.23 ± 0.04	P < 0.001 (all metrics)
Narrow Alley Passing	ML-MOPP	29.4 ± 1.8	0.15 ± 0.02	0.09 ± 0.01	-
	ChauffeurNet	33.7 ± 2.9	0.26 ± 0.05	0.19 ± 0.03	P < 0.001 (all metrics)

Table 6 presents a comparison of the generalization ability of our proposed model and several baselines. The results show that our model maintains the lowest curvature change (less than 0.181 per meter) and the smallest lateral offset in all scenarios.

Furthermore, the results indicate that traditional models such as RIT often exhibit looping behavior in roundabout scenarios. The comparison demonstrates that our proposed model possesses strong path planning capabilities.

Table 7. Results of ablation experiment

Model variants	ADE (m) ↓	FDE (m) ↓	Collision rate (%)↓	Planning success rate (%)↑	Discomfort level ↓
ML-MOPP (Complete Model)	0.70 ± 0.08	1.00 ± 0.12	1.1	95.6	0.28
- witho Meta-Imitation	1.05 ± 0.15	1.52 ± 0.22	2.8	89.3	0.35
- w/o GATr (spacetime graph)	1.22 ± 0.18	1.85 ± 0.31	4.5	85.1	0.41
- with/o CBF safety fixes	0.86 ± 0.11	1.31 ± 0.18	8.7	82.4	0.31
- Quantification of uncertainty with/out	0.73 ± 0.09	1.05 ± 0.13	1.9	92.8	0.30
- Quadratic Planner (Base Optimizer)	2.15 ± 0.40	3.20 ± 0.55	12.3	70.5	0.65

The meanings of the ablation variants are as follows: "Remove Meta-Imitation Learning" replaces it with standard pre-training + fine-tuning; "Remove Graph Attention Network" replaces it with a regular MLP; "Remove CBF Security Correction" removes security constraints; "Remove Uncertainty Quantization" removes the Bayesian module; and "Use Quadratic Planner Only" completely removes the learning module. All variants are evaluated on the same test set.

Table 7 compares the performance of the complete ML-MOPP model and its variants with key components removed on the same test set (100 complex urban scenes) to quantify the contribution of each module. Ablation experiments quantified the contribution of each module.

Removing meta-imitation learning led to a sharp increase in trajectory error, validating its core role in efficient data learning. Removing GATr increased the collision rate to 4.5%, indicating its criticality for modeling complex interactions. Removing CBF safety correction caused the collision rate to surge to 8.7%, highlighting its importance as a safety baseline. In contrast, removing the uncertainty quantization module primarily affected the collision rate (from 1.1% to 1.9%), with a smaller impact on trajectory accuracy. In conclusion, each module significantly contributes to the final performance of the model, collectively forming an efficient, safe, and robust system.

5. Conclusions

The ML-MOPP model proposed in this paper, through a two-stage architecture of "learning generation-optimization verification," has preliminarily verified its effectiveness in path planning tasks in 100 complex urban scenarios under the CARLA simulation environment. Experimental results show that compared with several baseline methods, the model exhibits significant advantages in multiple indicators such as trajectory accuracy (ADE 0.70 meters), planning success rate (95.6%), and passenger discomfort index (0.28). The model demonstrates its potential to rapidly adapt to new tasks in scenarios with small sample sizes (10 expert trajectories) through a meta-imitation learning mechanism. These results preliminarily demonstrate that a hybrid path planning method combining a data-driven generation module with a CBF-based safety verification module is a promising direction for addressing the trade-off between safety and generalization in autonomous driving.

While the simulation results are promising, the transfer from simulation to real-world deployment presents non-trivial challenges. The primary obstacles include the domain gap between simulated and real sensor modalities (LiDAR/camera), and the increased inference latency of the Transformer-based modules on resource-constrained embedded platforms such as NVIDIA Xavier. To bridge this gap, future work will focus on two complementary strategies: (i) incorporating domain randomization and adversarial training during the meta-training phase to enhance perceptual robustness, and (ii) implementing knowledge distillation and structured pruning to compress the GATr model, thereby satisfying real-time constraints without significantly compromising planning accuracy.

References

[1] Zhang, L. X., Yang, C., Yan, Y., Cai, Z., & Hu, Y. G. (2024). Automated guided vehicle dispatching and routing integration via digital twin with deep reinforcement learning. *Journal of Manufacturing Systems*, 72, 492–503. https://doi.org/10.1016/j.jmsy.2023.12.008

[2] Chen, Z. X., Liu, Y. Q., Ni, W. J., Hai, H., Huang, C. S., Xu, B. Y., et al. (2025). Predicting driving comfort in autonomous vehicles using road information and multi-head attention models. Nature Communications, 16(1), 14. https://doi.org/10.1038/s41467-025-57845-z

[3] Lv, Y. K., Chen, Y., Chen, Z. G., Fan, Y. Z., Tao, Y. C., Zhao, R., et al. (2025). Hybrid supervised and reinforcement learning for motion-sickness-aware path tracking in autonomous vehicles. *Sensors*, 25(12), 3695. https://doi.org/10.3390/s25123695

[4] Wang, S. C., Li, Z. B., Wang, B. T., & Li, M. (2024). Collision avoidance motion planning for connected and automated vehicle platoon merging and splitting with a hybrid automaton architecture. *IEEE Transactions on Intelligent Transportation Systems*, 25(2), 1445–1464. https://doi.org/10.1109/tits.2023.3315063

[5] Kim, D. J., Jeong, Y. W., & Chung, C. C. (2023). Lateral vehicle trajectory planning using a model predictive control scheme for an automated perpendicular parking system. *IEEE Transactions on Industrial Electronics*, 70(2), 1820–1829. https://doi.org/10.1109/tie.2022.3163567

[6] Cheng, S., Peng, H. N., Dong, X. X., Liu, Z. J., & Li, L. (2024). Chassis global dynamics-oriented trajectory planning for automated vehicles. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 54(2), 950–959. https://doi.org/10.1109/tsmc.2023.3321881

[7] Corno, M., Gimondi, A., Panzani, G., Roselli, F., Alessandretti, A., & Savaresi, S. M. (2023). A non-optimization-based dynamic path planning for autonomous obstacle avoidance. *IEEE Transactions on Control Systems Technology*, 31(2), 722–734. https://doi.org/10.1109/tcst.2022.3196880

[8] Lee, D., Kang, Y. S., & Noh, S. D. (2026). Digital twin-driven deep reinforcement learning for real-time optimisation in dynamic AGV systems. International *Journal of Production Research*, 64(1), 106–124. https://doi.org/10.1080/00207543.2025.2543491

[9] Typaldos, P., Papageorgiou, M., & Papamichail, I. (2022). Optimization-based path-planning for connected and non-connected automated vehicles. *Transportation Research Part C: Emerging Technologies*, 134, 103487. https://doi.org/10.1016/j.trc.2021.103487

[10] Zuo, T. L., Liu, H. K., Yang, S. C., Wang, W. Y., Peng, Y., & Wang, R. C. (2025). A reinforcement learning method for automated guided vehicle dispatching and path planning considering charging and path conflicts at an automated container terminal. *Journal of Marine Science and Engineering*, 14(1), 55. https://doi.org/10.3390/jmse14010055

[11] Fehér, A., Domina, A., Bárdos, A., Aradi, S., & Bécsi, T. (2025). Path planning via reinforcement learning with closed-loop motion control and field tests. *Engineering Applications of Artificial Intelligence*, 142, 109870. https://doi.org/10.1016/j.engappai.2024.109870

[12] Shu, Y. M., Zhou, J. Y., & Zhang, F. (2025). Agile decision-making and safety-critical motion planning for emergency autonomous vehicles. *IEEE Transactions on Intelligent Transportation Systems*, 26(9), 13750–13766. https://doi.org/10.1109/tits.2025.3569416

[13] Rahman, M. H., Gulzar, M. M., Haque, T. S., Habib, S., Shakoor, A., & Murtaza, A. F. (2025). Trajectory planning and tracking control in autonomous driving system: Leveraging machine learning and advanced control algorithms. *Engineering Science and Technology*, 64, 101950. https://doi.org/10.1016/j.jestch.2025.101950

[14] Chen, J., Pan, Y. X., Li, W. H., Lv, Z. Z., & Bai, X. X. (2026). A novel calibration method of driving safety field model based on traffic evolution scenarios with the application on path planning. *International Journal of Automotive Technology*, 27, 687–699. https://doi.org/10.1007/s12239-025-00397-4

[15] Ghajar, M., Alirezaei, M., Besselink, I., & Nijmeijer, H. (2024). A fast analytical local path planning method with applications in parking scenarios. *Proceedings of the Institution of Mechanical Engineers, Part D: Journal of Automobile Engineering*, 238(7), 2012–2026. https://doi.org/10.1177/09544070231154457

[16] Kim, D., Kim, G., Kim, H., & Huh, K. (2022). A hierarchical motion planning framework for autonomous driving in structured highway environments. *IEEE Access*, 10, 20102–20117. https://doi.org/10.1109/access.2022.3152187

[17] Dabestani, N., Typaldos, P., Yanumula, V. K., Papageorgiou, M., & Papamichail, I. (2024). Joint path planning for multiple automated vehicles in lane-free traffic with vehicle nudging. *IEEE Transactions on Intelligent Transportation Systems*, 25(11), 18525–18536. https://doi.org/10.1109/tits.2024.3445501

[18] Meng, T. C., Yang, T. H., Huang, J., Jin, W. R., Zhang, W., Jia, Y. F., et al. (2023). Improved hybrid a-star algorithm for path planning in autonomous parking system based on multi-stage dynamic optimization. *International Journal of Automotive Technology*, 24(2), 459–468. https://doi.org/10.1007/s12239-023-0038-1

[19] Zhang, B., Li, H. R., & Zheng, S. F. (2025). Enhanced automated valet parking system utilizing high-definition mapping and loop closure detection. *IEEE Transactions on Intelligent Transportation Systems*, 26, 1–13. https://doi.org/10.1109/tits.2025.3552400

[20] Chi, H., Cai, P., Fu, D., Zhai, J., Zeng, Y., & Shi, B. (2024). Spatiotemporal-restricted A∗ algorithm as a support for lane-free traffic at intersections with mixed flows. *Green Energy and Intelligent Transportation*, 3(2), 100159. https://doi.org/10.1016/j.geits.2024.100159

[21] Chen, W., Peng, J., Ma, Y., He, H., Ren, T., & Wang, C. (2025). Eco-driving framework for hybrid electric vehicles in multi-lane scenarios by using deep reinforcement learning methods. *Green Energy and Intelligent Transportation*, 100309. https://doi.org/10.1016/j.geits.2025.100309

[22] Chi, H., Cai, P., Fu, D., Zhai, J., Zeng, Y., & Shi, B. (2024). Spatiotemporal-restricted A∗ algorithm as a support for lane-free traffic at intersections with mixed flows. Green Energy and Intelligent Transportation, 3(2), 100159. https://doi.org/10.1016/j.geits.2024.100159

[23] Chen, W., Peng, J., Ma, Y., He, H., Ren, T., & Wang, C. (2025). Eco-driving framework for hybrid electric vehicles in multi-lane scenarios by using deep reinforcement learning methods. Green Energy and Intelligent Transportation, 100309. https://doi.org/10.1016/j.geits.2025.100309

[24] Lin, Z. Y., Ma, J., Duan, J. L., Li, S. E., Ma, H. T., Cheng, B., et al. (2023). Policy iteration based approximate dynamic programming toward autonomous driving in constrained dynamic environment. *IEEE Transactions on Intelligent Transportation Systems*, 24(5), 5003–5013. https://doi.org/10.1109/tits.2023.3237568

[25] Jin, X. J., Tao, Y. C., & Ikiela, N. V. O. (2024). Trajectory planning design for parallel parking of autonomous ground vehicles with improved safe travel corridor. *Symmetry*, 16(9), 1129. https://doi.org/10.3390/sym16091129

[26] Cai, M. C., Xu, Q., Chen, C. Y., Wang, J. W., Li, K. Q., Wang, J. Q., et al. (2023). Formation control for connected and automated vehicles on multi-lane roads: Relative motion planning and conflict resolution. *IET Intelligent Transport Systems*, 17(1), 211–226. https://doi.org/10.1049/itr2.12249

[27] Tang, Q., & Wang, H. (2025). Data-driven automated job shop scheduling optimization considering AGV obstacle avoidance. *Scientific Reports*, 15(1), 19. https://doi.org/10.1038/s41598-024-82870-1