MFPAD: Memory–Forgetting Planning for Long-Horizon End-to-End Autonomous Driving

Yikai Wu, Qizhou Hu, Aiguo Lei, and Ziying Song

***Abstract*—Recent planning-oriented end-to-end autonomous driving methods have achieved competitive performance on short-horizon (3 s) trajectory prediction. However, their accuracy and stability degrade markedly when the prediction horizon is extended to long-horizon (6 s), leading to drifting trajectories. A key reason is that most existing frameworks adopt simple feed-forward multilayer perceptron regressors in the trajectory refinement head, which lack explicit modeling of long-term temporal dependencies and planning inertia. To address this limitation, we propose the Memory–Forgetting Planning for Autonomous Driving, a plug-in refinement head that combines an LSTM-based memory network with a Transformer-based forgetting network. The memory network autoregressively rolls out a coarse long-horizon trajectory and exposes a sequence of hidden states, while the forgetting network attends over these states and surrounding-agent features with token-level dropout to suppress outdated or noisy motion cues. A lightweight gating module fuses coarse and corrected trajectories at each time step, yielding temporally consistent, interaction-aware plans over a 6 s horizon. We evaluate our method on NuScenes, Adv-NuScenes, Bench2Drive, and NAVSIM, and the results demonstrate consistent improvements. Compared with baselines, it reduces the average collision rate on NuScenes by 11.1% and the 3 s collision rate on Adv-NuScenes by 29.2%, while achieving a 6 s collision rate of 2.01% on NuScenes. In addition, the closed-loop results on Bench2Drive and NAVSIM show that the proposed refinement head also improves downstream driving performance under feedback-driven evaluation. The source code is available at https://github.com/Y1Ka1/MFPAD.**

Link to graphical and video abstracts, and to code:
https://latamt.ieeer9.org/index.php/transactions/article/view/10554

***Index Terms*—End-to-End Autonomous Driving, Trajectory Prediction, Motion Planning, Long Horizon.**

I. Introduction

In recent years, autonomous driving systems have increasingly moved from handcrafted modular perception–prediction–planning pipelines toward *end-to-end autonomous driving* (E2E-AD) frameworks. Instead of executing perception, prediction, and planning as loosely coupled stages, E2E-AD approaches employ unified differentiable architectures that map multi-view sensor streams directly to future trajectories or low-dimensional control commands, thereby reducing error propagation between modules and enabling joint optimization of driving behavior [1]–[5]. In parallel, the robotics and control communities have developed extensive work on model-based trajectory generation and tracking for mobile robots and autonomous vehicles, including Bézier-polynomial trajectory generation for wheeled robots, sliding-mode trajectory tracking for aerial vehicles, and low-cost experimental testbeds for autonomous cars [6]–[13].

Building on these foundations, recent planning-oriented E2E-AD frameworks such as interpretable neural motion planners and unified map–perceive–predict–plan architectures [14]–[18], as well as large-scale systems like UniAD, VAD, SparseDrive, and MomAD [19]–[22], have demonstrated competitive performance on benchmarks such as NuScenes [23], particularly for short-horizon trajectory prediction and planning in complex urban traffic. More recently, generative planning heads based on diffusion, flow matching, and local motion modeling have been proposed to better capture multi-modal future trajectories and improve robustness on NAVSIM, NuScenes, Bench2Drive, and Adv-NuScenes, including DiffusionDrive, Diver, GuideFlow, GoalFlow and FocalAD [24]–[29]. At the same time, recent closed-loop benchmarks such as NAVSIM and Bench2Drive have also motivated a broader range of planning-oriented E2E-AD methods, including diffusion-based or hybrid approaches such as DiffE2E and BridgeDrive, as well as trajectory-selection and expert-distillation frameworks such as DriveSuprim and Hydra-MDP++ [30]–[33].

While these planning-oriented E2E-AD frameworks have achieved substantial progress, their trajectory planners are predominantly optimized and evaluated over short prediction horizons, and even recent generative planners based on diffusion or flow matching typically adopt the same 2–3 s-oriented evaluation protocol on NuScenes-like benchmarks [24]–[27]. In UniAD, VAD, SparseDrive, and MomAD, the ego-trajectory head is typically implemented as a lightweight feed-forward regressor and benchmarked mainly within a 2–3 s window on NuScenes, where local scene geometry and near-future intentions dominate [19]–[23]. As a consequence, the behavior of these models beyond 3 s—in particular at 5–6 s horizons—is seldom analyzed in detail and is not explicitly modeled in their architectural design, even though the dataset provides annotations over a 6 s future. In real traffic, however, a 6 s look-ahead horizon is critical for anticipatory decision making: highway lane changes, merges at on-ramps, unprotected turns at intersections, and interactions with aggressive or hesitant agents all require the ego vehicle to reason about how its actions will unfold several seconds ahead, not just in the next few meters. Even under frequent replanning, short-horizon planners can still exhibit myopic behavior, leading to oscil-

The associate editor coordinating the review of this manuscript and approving it for publication was Alejandro Dzul (*Corresponding author: Yikai Wu*).

Yikai Wu, Q. Hu, and A. Lei are with Nanjing University of Science and Technology, China (e-mails: wuyikai@njust.edu.cn, qizhouhu@163.com, and leiaiguo9601@163.com).

Z. Song is with Beijing Jiaotong University, China (e-mail: 22110110@bjtu.edu.cn).

latory trajectories, late braking, or uncomfortable jerk when confronted with long-range conflict scenarios. In this sense, our 6 s long-horizon design is not intended to replace high-frequency replanning, but to improve the quality, consistency, and anticipatory capability of each predicted planning horizon on which such replanning operates. Moreover, downstream tracking controllers and safety verifiers operate on the entire planned horizon; unstable or drifting predictions at 5–6 s increase the burden on low-level control and undermine the reliability of closed-loop evaluation. This motivates us to explicitly focus on *long-horizon* (6 s) trajectory prediction and to validate the resulting planning head not only in open-loop settings, but also under closed-loop driving benchmarks. Accordingly, our goal is to improve long-horizon planning stability through trajectory refinement, rather than to treat long-term behavior as a byproduct of short-horizon optimization or short-horizon generative sampling. To further probe robustness under intentionally challenging interactions, we additionally consider Adv-NuScenes, an adversarial extension of NuScenes constructed by the Challenger framework [34].

However, in most of these frameworks the strong spatiotemporal encoders are ultimately coupled with a simple trajectory head: a feed-forward multi-layer perceptron (MLP) that regresses all future waypoints from a single planning query [19]–[22], [35]. Such MLP heads are straightforward to optimize around the 2–3 s window, but they treat different time steps as independent outputs and lack any explicit notion of temporal memory or planning inertia. As a result, short-horizon predictions remain accurate whereas the 5–6 s portion of the trajectory often drifts, oscillates, or deviates from the intended lane geometry, as illustrated in Fig. 1. This suggests that, in many cases, the bottleneck for long-horizon performance lies less in perception and more in the design of the trajectory refinement module itself. Although recurrent and attention-based architectures have been studied in broader sequence modeling settings, they are typically introduced for generic temporal representation learning or full sequence decoding, rather than for long-horizon trajectory refinement in planning-oriented E2E-AD. A desirable long-horizon planning head should therefore satisfy two requirements: it should preserve temporal continuity and planning inertia over future rollout, while remaining adaptable as the horizon extends and accumulated motion cues gradually become stale or noisy. Our goal is therefore not to introduce another general temporal backbone, but to design a task-specific refinement module for improving long-horizon ego planning.

To address these limitations, we propose a *Memory–Forgetting Planning Network* (MFPAD), which serves as a plug-in trajectory refinement head (MFPAD head) for planning-oriented E2E-AD models. Concretely, MFPAD replaces the conventional MLP trajectory head with a two-stage memory–forgetting architecture: an LSTM-based memory network that autoregressively rolls out a coarse long-horizon trajectory and exposes a sequence of hidden states, and a Transformer-based forgetting network that attends over these states with token-level dropout to filter outdated or noisy motion cues. A lightweight gating module then fuses the coarse and corrected trajectories at each future step, yielding temporally consistent long-horizon plans over the 6 s horizon. We conduct comprehensive experiments on the NuScenes benchmark [23], an adversarially perturbed variant (Adv-NuScenes) [34], and two closed-loop benchmarks, Bench2Drive and NAVSIM. On NuScenes, MFPAD achieves an average L2 error of 1.19 m and an average collision rate of 0.80% over 1–6 s, with a 6 s collision rate of 2.01%. Compared with MomAD [22], MFPAD reduces the 1–6 s average L2 by 15.6% (1.41 $\rightarrow$ 1.19 m). On Adv-NuScenes, MFPAD further improves robustness-related metrics, while the closed-loop results on Bench2Drive and NAVSIM show that the proposed refinement head also improves downstream driving performance under feedback-driven evaluation.

In summary, the main contributions of this work are as follows:

1) We revisit planning-oriented end-to-end autonomous driving from a long-horizon perspective and explicitly study 6 s trajectory prediction, where conventional MLP-based refinement heads exhibit evident drift and instability.
2) We propose MFPAD, a plug-in memory–forgetting planning head that performs long-horizon refinement in a coarse-to-correct manner: an LSTM-based memory branch first rolls out a coarse trajectory, and a lightweight Transformer-based forgetting branch then selectively suppresses stale or noisy temporal cues via token-level dropout and gated correction.
3) We conduct extensive experiments on NuScenes, Adv-NuScenes, Bench2Drive, and NAVSIM, showing that MFPAD consistently improves long-horizon planning accuracy, yields better robustness-related metrics under perturbation, and also improves closed-loop driving performance under feedback-driven evaluation.

II. Method

A. Framework Overview

Our framework builds upon the unified motion–planning architecture of MomAD [22], which takes surround-view multi-camera images as input, encodes them into scene features, and jointly reasons about dynamic agents, map semantics, and ego motion through a transformer-based encoder and a multi-branch refinement head. In our implementation, we follow this camera-based pipeline and keep the perception backbone, sparse scene representation, and momentum planning modules of MomAD unchanged. In the original MomAD, both the motion and planning refinement branches are implemented as lightweight multilayer perceptron (MLP) regressors. Given a planning query embedding $\mathbf{q}_t \in \mathbb{R}^D$ at time step t, the MLP takes $\mathbf{q}_t$ as input and directly outputs all future ego waypoints as a single vector, which is then reshaped into a sequence of T two-dimensional coordinates. In other words, all T future steps are produced in one shot by a stack of fully connected layers, without any explicit temporal recurrence or attention along the prediction horizon. While this design is computationally efficient and achieves satisfactory performance for short-horizon prediction (e.g., 2–3 s), it does not encode temporal dependencies or long-term planning uncertainty, and empirically tends

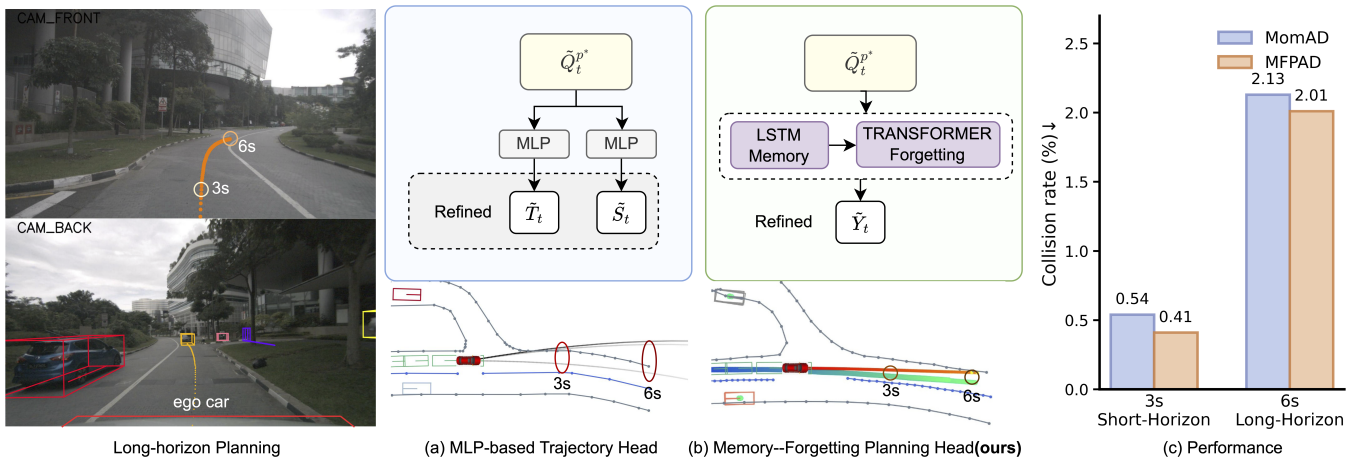

Fig. 1. Motivation for long-horizon planning. (a) MLP-based refinement heads in existing E2E-AD planners produce accurate 3 s trajectories but drift at 6 s. (b) The proposed Memory–Forgetting Planning Network (MFPAD) yields temporally consistent 6 s trajectories. (c) Quantitative comparison in terms of long-horizon displacement error, collision rate, and trajectory prediction consistency (TPC).

to exhibit drift and inconsistent behavior when extended to 5–6 s horizons.

To address these limitations, we introduce the **Memory–Forgetting Planning Network (MFPAD)**, a task-specific trajectory refinement module for long-horizon ego planning. The design follows a coarse-to-correct perspective: a desirable long-horizon planning head should preserve temporal continuity during future rollout, while remaining able to correct accumulated stale or noisy cues as the horizon extends. Accordingly, MFPAD consists of a *Memory Network* for temporally coherent coarse rollout and a *Forgetting Network* for selective refinement and stabilization. Rather than treating long-horizon planning as direct sequence regression, MFPAD organizes the prediction process as memory-conditioned trajectory refinement. The memory branch performs query-conditioned autoregressive rollout to produce both a coarse trajectory prior and a time-indexed memory bank; the forgetting branch refines this memory through future-step queries and predicts residual corrections; and the gated fusion module performs step-wise trajectory-space trust allocation between the preserved memory and the learned correction. Thus, MFPAD is organized as a functional decomposition of long-horizon refinement into continuity-preserving rollout and correction-oriented forgetting, rather than as a generic sequence head. A structured summary of the proposed architecture is provided in Table I.

B. Memory Network

Given a set of planning query embeddings $\tilde{\mathbf{Q}}^{p*} = \{\tilde{\mathbf{q}}_i\}_{i=1}^{N}$ produced by the planning head, we first aggregate them into a single global query $\mathbf{q}_{\text{global}} \in \mathbb{R}^{D_q}$ using their classification logits as attention weights:

$$\mathbf{q}_{\text{global}} = \sum_{i=1}^{N} \pi_i \tilde{\mathbf{q}}_i, \qquad \pi_i = \frac{\exp(s_i)}{\sum_{j=1}^{N} \exp(s_j)}, \tag{1}$$

where s_i denotes the logit of the i-th planning mode and $\{\pi_i\}$ are the corresponding normalized weights. We then project $\mathbf{q}_{\text{global}}$ into the memory space to initialize the hidden and cell states of an LSTM:

$$\mathbf{h}_0 = \tanh(\mathbf{W}_h \mathbf{q}_{\text{global}}), \quad \mathbf{c}_0 = \tanh(\mathbf{W}_c \mathbf{q}_{\text{global}}), \tag{2}$$

where $\mathbf{W}_h, \mathbf{W}_c \in \mathbb{R}^{D_m \times D_q}$ and D_m is the hidden dimension of the memory network. The LSTM is then applied autoregressively to generate coarse future positions:

$$(\mathbf{h}_t, \mathbf{c}_t) = \text{LSTMCell}\big([\mathbf{y}_{t-1}, \mathbf{q}_{\text{global}}], (\mathbf{h}_{t-1}, \mathbf{c}_{t-1})\big), \tag{3}$$

$$\Delta \mathbf{y}_t = \mathbf{W}_o \mathbf{h}_t, \quad \mathbf{y}_t = \mathbf{y}_{t-1} + \Delta \mathbf{y}_t, \tag{4}$$

where $\mathbf{y}_t \in \mathbb{R}^2$ represents the 2D position at step t. This recurrent process provides a temporally coherent coarse trajectory $\mathbf{Y}^{\text{mem}}$ that serves as a motion prior for subsequent refinement.

C. Forgetting Network

Although the LSTM-based memory branch captures long-range temporal dependencies, its hidden states may accumulate outdated or noisy motion cues over long horizons. To selectively refine these representations, we introduce a Transformer-based *Forgetting Network* that operates on the memory states and, optionally, surrounding-instance context.

Let the LSTM hidden states be

$$H = [\mathbf{h}_1, \mathbf{h}_2, \ldots, \mathbf{h}_T] \in \mathbb{R}^{T \times D_m}, \tag{5}$$

and let $F_t^{\text{ins}} \in \mathbb{R}^{M \times D_{\text{ins}}}$ denote instance features of surrounding agents or map elements at the current frame. We first project both memory and instance features into a shared token space:

$$\mathbf{M}_{\text{mem}} = f_{\text{mem}}(H) \in \mathbb{R}^{T \times D}, \tag{6}$$

$$\mathbf{M}_{\text{ins}} = f_{\text{ins}}(F_t^{\text{ins}}) \in \mathbb{R}^{M \times D}, \tag{7}$$

where M is the number of selected instance features at the current frame. For efficiency, we only keep the top K instance tokens (e.g., K=32) in practice, and when instance features are unavailable we simply set $\mathbf{M}_{\text{ins}}$ to empty. We then concatenate the two sets of tokens along the token dimension to form the key–value memory:

$$\mathbf{M} = [\mathbf{M}_{\text{mem}}; \mathbf{M}_{\text{ins}}] \in \mathbb{R}^{(T+M) \times D}. \tag{8}$$

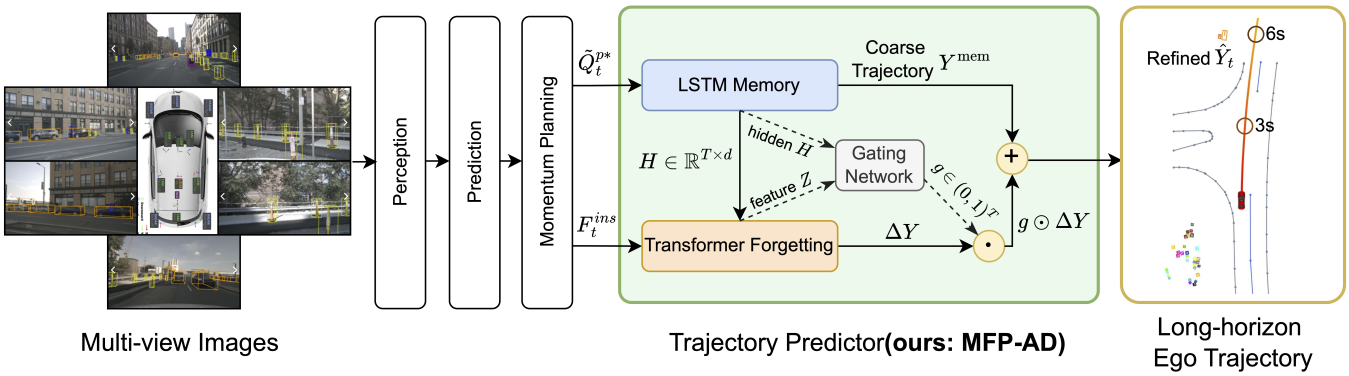

Fig. 2. Overall architecture of the proposed method. We follow MomAD for BEV encoding, sparse perception, and momentum planning (left), and replace the original MLP-based trajectory predictor with a Memory–Forgetting Planning Network (MFPAD, right). The LSTM-based memory branch produces a coarse long-horizon trajectory Y^{mem} and hidden states H, while the Transformer-based forgetting branch computes a correction ΔY from H and instance features F_t^{ins}. A gating network outputs per-step weights $g \in (0,1)^T$ to scale the correction, and the final refined ego trajectory is obtained as $\hat{Y} = Y^{\text{mem}} + g \odot \Delta Y$.

TABLE I

ARCHITECTURE SUMMARY OF THE PROPOSED MFPAD HEAD. WE SUMMARIZE THE MAIN STAGES, CONFIGURATIONS, TENSOR SHAPES, INPUTS/OUTPUTS, AND THEIR FUNCTIONAL ROLES

Stage	Module	Configuration	Tensor shape	Input / Output	Role
Query initialization	Planning query aggregation	Inherited from MomAD; query dimension D_q ($D_q = 256$ in our implementation)	$\tilde{\mathbf{Q}}^{p*} : N \times D_q$ $\mathbf{q}_{\text{global}} : D_q$	Input: planning query embeddings $\tilde{\mathbf{Q}}^{p*} = \{\tilde{\mathbf{q}}_i\}_{i=1}^{N}$ Output: global planning query $\mathbf{q}_{\text{global}}$	Aggregates the planning queries from the MomAD planning branch to form the initial ego-planning representation for trajectory refinement.
Coarse rollout	Memory Network	LSTM; hidden size D_m ($D_m = 256$ in our implementation); autoregressive horizon $T = 6$ s	$\mathbf{H} : T \times D_m$ $\mathbf{Y}^{\text{mem}} : T \times 2$	Input: $\mathbf{q}_{\text{global}}$ Output: coarse trajectory $\mathbf{Y}^{\text{mem}}$ and hidden states $\mathbf{H} = \{\mathbf{h}_1, \ldots, \mathbf{h}_T\}$	Performs recurrent long-horizon rollout to preserve temporal continuity and planning inertia.
State refinement	Forgetting Network	Transformer decoder; $L = 2$ layers; 4 attention heads; model dimension D ($D = 256$ in our implementation); token dropout rate $p_f = 0.2$	$\mathbf{H} : T \times D_m$ $\mathbf{F}_t^{\text{ins}} : M \times D_{\text{ins}}$ $\mathbf{Z} : T \times D$	Input: memory states $\mathbf{H}$ and optional instance context $\mathbf{F}_t^{\text{ins}}$ Output: refined per-step features $\mathbf{Z} = \{\mathbf{z}_1, \ldots, \mathbf{z}_T\}$	Selectively refines the accumulated memory states by suppressing stale or noisy cues.
Correction prediction	Trajectory correction head	Linear prediction layer	$\mathbf{Z} : T \times D$ $\Delta\mathbf{Y} : T \times 2$	Input: refined features $\mathbf{Z}$ Output: trajectory correction $\Delta\mathbf{Y}$	Maps the refined features to step-wise trajectory corrections.
Fusion	Gated fusion module	Lightweight gating function	$\mathbf{g} : T$ $\hat{\mathbf{Y}} : T \times 2$	Input: coarse trajectory $\mathbf{Y}^{\text{mem}}$, correction $\Delta\mathbf{Y}$, memory states $\mathbf{H}$, and refined features $\mathbf{Z}$ Output: gate $\mathbf{g}$ and final trajectory $\hat{\mathbf{Y}}$	Balances continuity preservation and correction at each future step through gated fusion.

To provide time-aware queries, we introduce a learnable matrix of temporal query embeddings

$$\mathbf{Q}_{\text{time}} = [\mathbf{q}_1, \ldots, \mathbf{q}_T] \in \mathbb{R}^{T \times D}, \tag{9}$$

where each $\mathbf{q}_t$ corresponds to a future planning step. The Transformer decoder attends from these time queries to the memory tokens:

$$\mathbf{Z} = \text{TransformerDecoder}(\mathbf{Q}_{\text{time}}, \tilde{\mathbf{M}}), \tag{10}$$

where $\mathbf{Z} = [\mathbf{z}_1, \ldots, \mathbf{z}_T] \in \mathbb{R}^{T \times D}$ are refined per-step features for trajectory correction.

To explicitly implement forgetting and improve robustness, we apply token-level dropout on the memory tokens during training. Let $\mathbf{R} \in \{0,1\}^{(T+M)\times 1}$ be a random mask, whose entries are drawn from a Bernoulli distribution with keep probability $1 - p_f$:

$$\mathbf{R}_i \sim \text{Bernoulli}(1 - p_f), \quad i = 1, \ldots, T + M. \tag{11}$$

The effective memory tokens fed into the decoder are then given by

$$\tilde{\mathbf{M}} = \mathbf{M} \odot \mathbf{R}, \tag{12}$$

where $\odot$ denotes element-wise multiplication with broadcasting over the feature dimension. This token-level operation simulates partial forgetting by applying structured perturbation to the memory/context tokens before refinement, rather than by dropping generic decoder features. The subsequent Transformer decoder and gated fusion module then learn to correct the trajectory under incomplete or noisy memory conditions. Therefore, the adaptive behavior of the forgetting process comes from the conditional refinement and trajectory-space gating response, rather than from the random mask alone.

D. Gated Fusion of Memory and Correction

The forgetting network produces refined features $\mathbf{Z}$ that encode long-horizon context and agent interactions. On top

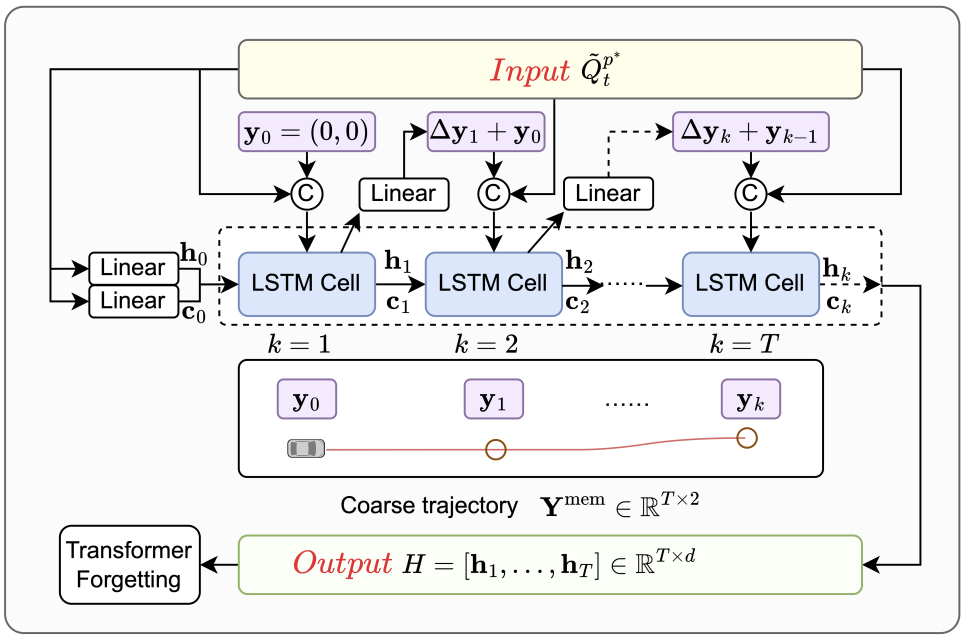

Fig. 3. LSTM-based memory branch for long-horizon planning. For each frame, the planning head outputs a set of query embeddings $\tilde{\mathbf{Q}}^{p*} = \{\tilde{\mathbf{q}}_i\}_{i=1}^{N}$. We aggregate them into a global planning query $\mathbf{q}_{\text{global}}$ and project it by two linear layers to initialize the hidden and cell states $(\mathbf{h}_0, \mathbf{c}_0)$. Starting from the ego-centric origin $\mathbf{y}_0 = (0,0)$, at each time step k the LSTM cell takes as input the concatenation of the previous position $\mathbf{y}_{k-1}$ and $\mathbf{q}_{\text{global}}$, and updates $(\mathbf{h}_k, \mathbf{c}_k)$. A linear head predicts an incremental offset $\Delta\mathbf{y}_k$, which is accumulated as $\mathbf{y}_k = \mathbf{y}_{k-1} + \Delta\mathbf{y}_k$ to form the coarse trajectory $\mathbf{Y}^{\text{mem}} \in \mathbb{R}^{T\times 2}$. The hidden sequence $H = [\mathbf{h}_1, \ldots, \mathbf{h}_T] \in \mathbb{R}^{T\times d}$ is passed to the Transformer-based forgetting branch.

of $\mathbf{Z}$, we predict step-wise trajectory corrections and learn a gating mechanism to balance the new corrections against the coarse memory trajectory from the LSTM branch.

First, a linear head maps the decoded features into 2D trajectory offsets:

$$\Delta\mathbf{y}_t = W_{\text{corr}}\mathbf{z}_t + \mathbf{b}_{\text{corr}}, \qquad \Delta\mathbf{Y} = [\Delta\mathbf{y}_1, \ldots, \Delta\mathbf{y}_T] \in \mathbb{R}^{T\times 2}, \tag{13}$$

where $\Delta\mathbf{y}_t$ represents a fine-grained correction to the coarse position at time step t.

At the same time, we compute a forgetting gate g_t for each future step by combining the LSTM memory state and the decoded feature. Let $\tilde{\mathbf{h}}_t = f_{\text{proj}}(\mathbf{h}_t)$ be a projected memory feature, and define

$$\mathbf{u}_t = [\tilde{\mathbf{h}}_t, \mathbf{z}_t], \tag{14}$$

where $[\cdot,\cdot]$ denotes concatenation. The gate is obtained via a small MLP followed by a sigmoid:

$$g_t = \sigma\big(f_{\text{gate}}(\mathbf{u}_t)\big), \qquad \mathbf{g} = [g_1, \ldots, g_T] \in (0,1)^T, \tag{15}$$

where g_t measures how much the planner should trust the newly computed correction at step t.

Given the coarse trajectory from the memory branch,

$$\mathbf{Y}^{\text{mem}} = [\mathbf{y}_1^{\text{mem}}, \ldots, \mathbf{y}_T^{\text{mem}}] \in \mathbb{R}^{T\times 2}, \tag{16}$$

the final refined planning trajectory is computed as

$$\hat{\mathbf{Y}} = \mathbf{Y}^{\text{mem}} + \mathbf{g} \odot \Delta\mathbf{Y}, \tag{17}$$

where the gate vector $\mathbf{g}$ is broadcast over the 2D coordinates. This formulation allows the planner to dynamically balance long-term memory and short-term corrections: when g_t is small, the model relies more on the memorized trend; when g_t is large, it strongly adapts to the refined corrections, effectively realizing *learned memory and adaptive forgetting* for long-horizon planning.

E. Training and Integration

The proposed refinement module is differentiable and can be integrated into the MomAD training pipeline. The overall optimization objective follows the standard multi-task loss formulation:

$$\mathcal{L}_{\text{total}} = \lambda_{\text{cls}}\mathcal{L}_{\text{CE}} + \lambda_{\text{reg}}\mathcal{L}_{\text{SmoothL1}} + \lambda_{\text{status}}\mathcal{L}_{\text{ego}}, \tag{18}$$

where $\mathcal{L}_{\text{reg}}$ supervises the predicted trajectories $\hat{\mathbf{y}}_t$ against ground truth ego trajectories. The joint optimization of the LSTM and Transformer components improves convergence stability and enhances temporal coherence without additional supervision.

III. Experiments

A. Datasets and Metrics

We evaluate the proposed Memory–Forgetting Planning (MFPAD) head on the NuScenes, Adv-NuScenes, Bench2Drive, and NAVSIM benchmarks.

NuScenes. NuScenes [23] is a large-scale autonomous driving dataset containing 1,000 scenes, each 20 s long, collected in diverse urban environments. Each scene is annotated at 2 Hz with 3D bounding boxes for vehicles, pedestrians, and traffic participants, together with accurate ego-pose and HD map information. The scenes are manually selected to cover a diverse

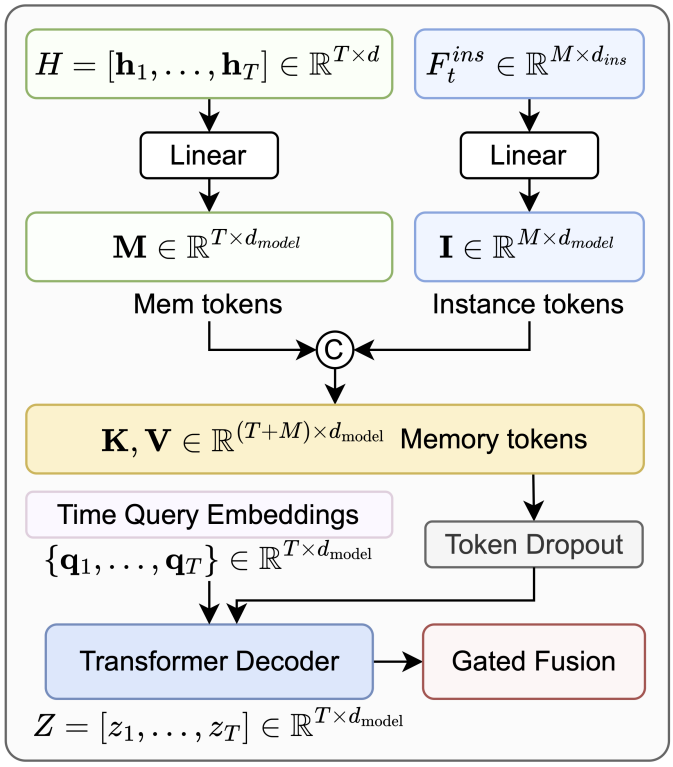

Fig. 4. Architecture of the Transformer-based forgetting network. The LSTM hidden sequence $H = [\mathbf{h}_1, \ldots, \mathbf{h}_T] \in \mathbb{R}^{T\times d}$ and, optionally, a set of surrounding-instance features $F_t^{\text{ins}} \in \mathbb{R}^{M\times d_{\text{ins}}}$ are projected into memory tokens $\mathbf{M} \in \mathbb{R}^{T\times d_{\text{model}}}$ and $\mathbf{I} \in \mathbb{R}^{M\times d_{\text{model}}}$, which are concatenated as keys/values $\mathbf{K}, \mathbf{V}$. Learnable time query embeddings $\{\mathbf{q}_1, \ldots, \mathbf{q}_T\} \in \mathbb{R}^{T\times d_{\text{model}}}$ serve as decoder queries, while token dropout is applied to memory tokens during training to simulate forgetting. The Transformer decoder outputs $\mathbf{Z} = [\mathbf{z}_1, \ldots, \mathbf{z}_T] \in \mathbb{R}^{T\times d_{\text{model}}}$, which is then fed into the gated fusion module to produce trajectory corrections.

and challenging set of driving maneuvers, traffic situations, and unexpected behaviors, making NuScenes a suitable benchmark for evaluating long-horizon ego planning in complex urban traffic. Following prior planning-oriented E2E-AD works such as UniAD, VAD, SparseDrive and MomAD [19]–[22], we use the standard NuScenes validation set for open-loop planning evaluation.

Adv-NuScenes. To analyze robustness under challenging interactions, we also use the Adv-NuScenes (Adv-nuSc) dataset generated by the Challenger framework of Xu *et al.* [34]. Adv-NuScenes is built by replaying NuScenes traffic scenes while injecting an adversarial vehicle whose trajectory is optimized to challenge the ego car (e.g., cut-ins, tailgating, blocking) but still remain physically plausible. The released Adv-nuSc split contains 156 scenes (6,115 samples), with multi-view images and BEV annotations rendered at the same frequency as the original NuScenes validation set. Compared with the original NuScenes setting, Adv-NuScenes increases interaction difficulty through diverse adversarial yet realistic behaviors across dynamic urban contexts, providing a more demanding benchmark for evaluating long-horizon planning robustness and safety.

Bench2Drive. To further evaluate closed-loop driving performance, we additionally use Bench2Drive [36], a closed-loop benchmark based on CARLA Leaderboard 2.0 for end-to-end autonomous driving. Following prior works, we use the official training data under the base setting (1000 clips) for fair comparison, and evaluate on the official 220 routes.

NAVSIM. We further evaluate MFPAD on NAVSIM [37], a data-driven non-reactive simulation benchmark for end-to-end autonomous driving. In this work, we report results on the navtest split following the official protocol.

Evaluation Metrics. We follow MomAD [22] and report three categories of open-loop planning metrics:

- *Displacement error* (L2, m): the average Euclidean distance between predicted and ground-truth waypoints at each forecast horizon $t \in \{1, \ldots, 6\}$ s.
- *Collision rate* (%): the percentage of predicted trajectories that collide with any dynamic or static object when rolled out in the BEV space up to horizon t.
- *Trajectory Prediction Consistency* (TPC, m) [22]: the mean discrepancy between consecutive re-planned trajectories, measuring temporal smoothness and planning stability.

For closed-loop evaluation, we additionally report the official benchmark metrics on Bench2Drive and NAVSIM. In Bench2Drive, we use Driving Score (DS), Success Rate (SR), Efficiency, Comfort, and Multi-Ability scores. In NAVSIM, we use No At-Fault Collision (NC), Drivable Area Compliance (DAC), Time-to-Collision (TTC), Comfort, Ego Progress (EP), and the Predictive Driver Model Score (PDMS). In addition, we report the runtime of different methods as frames per second (FPS) on the same hardware for a fair comparison. For clarity, we separately summarize *short-horizon* performance within 1–3 s and *long-horizon* performance up to 6 s, while closed-loop results are discussed separately on Bench2Drive and NAVSIM.

B. Implementation Details

Network configurations. Our implementation is based on the publicly released MomAD codebase. Unless otherwise specified, we follow the camera-based MomAD setting and keep the perception backbone, sparse BEV representation, and momentum planning modules identical to MomAD, while replacing only the trajectory predictor with our MFPAD head. No LiDAR or radar branch is introduced in our pipeline. The embedding dimension of the planning query is set to $d = 256$. The LSTM memory uses a hidden size of d and an autoregressive horizon of $T = 6$ s, that matches the ego-trajectory prediction horizon. The Transformer forgetting network adopts $L = 2$ decoder layers with 4 attention heads and a model dimension of 256. The token dropout rate on the forgetting network is set to $p_f = 0.2$.

Training setup. All models are trained using the AdamW optimizer with an initial learning rate of $5e-5$, weight decay of 0.001, and a cosine decay schedule over 20 epochs. The batch size is set to 8 frames per GPU. For a fair comparison, MomAD [22], SparseDrive [21], and our MFPAD head share the same data augmentations, training splits, and loss weights as in the original MomAD settings. When reporting results

on Adv-NuScenes, we fine-tune the planning head starting from the NuScenes checkpoint, while keeping the perception backbone frozen.

Hardware and software. All experiments are conducted on a cloud server running Ubuntu 20.04 with Python 3.8 and PyTorch 2.0. The machine is equipped with a single virtual GPU (vGPU) with 48 GB memory (NVIDIA L20 class), a 12-core Intel Xeon Gold 6459C CPU, and 72 GB system RAM. The CUDA version is 11.8. Training a MomAD baseline model for 20 epochs takes approximately 24 hours in this setup, and replacing the MLP head with the proposed MFPAD head yields comparable wall-clock time without a noticeable increase. In terms of deployment cost, MFPAD runs at 6.6 FPS on the evaluated hardware, compared with 7.8 FPS for MomAD. Since our modification is limited to the trajectory refinement head, this result indicates a modest runtime overhead relative to the base system on the evaluated hardware. This runtime difference is mainly introduced by the trajectory refinement head and represents a modest accuracy–efficiency trade-off for improved long-horizon planning stability.

C. Main Results on NuScenes

The main open-loop results on NuScenes are presented from three evaluation perspectives: the primary long-horizon setting over 1–6 s, an extended-horizon analysis beyond 6 s, and a short-horizon comparison over 1–3 s for consistency with prior planning-oriented E2E-AD methods.

1) Long-horizon Planning on NuScenes: Table II reports open-loop planning performance on the NuScenes validation set from 1 s to 6 s. We compare representative planning-oriented E2E-AD baselines (UniAD, SparseDrive, and MomAD) against our proposed MFPAD. To verify that the proposed refinement head is not tied to a single base architecture, we additionally integrate it into SparseDrive and compare the baseline with its MFPAD-enhanced counterpart under the same training and evaluation settings. For each method, we report the L2 displacement error and collision rate at multiple horizons, together with inference speed (FPS).

As shown in Table II, MFPAD achieves the lowest average L2 error (1.19 m) and the lowest average collision rate (0.80%) over 1–6 s. Compared with MomAD, MFPAD reduces the average L2 from 1.41 m to 1.19 m (15.6% relative reduction) and decreases the average collision rate from 0.90% to 0.80% (11.1% relative reduction), while maintaining an inference speed of 6.6 FPS on the evaluated hardware, with only a modest slowdown relative to the MomAD baseline (7.8 FPS). These gains are more pronounced at longer horizons: at 6 s, MFPAD lowers L2 from 2.45 m to 2.30 m (6.1% relative reduction) and reduces collision from 2.13% to 2.01% (5.6% relative reduction), indicating improved stability of long-horizon planning. Compared with DiffusionDrive, DIVER and Guideflow, the results indicate that the proposed memory–forgetting refinement is effective not only against conventional planning heads, but also against recent diffusion- and flow-based planners under the same long-horizon evaluation protocol. Moreover, the proposed head also improves SparseDrive consistently, reducing the average L2 from 1.59 m to 1.38 m (13.2% relative reduction) and the average collision rate from 0.97% to 0.88% (9.3% relative reduction). This indicates that the effectiveness of MFPAD is not limited to the MomAD base framework, but can generalize to another representative planning-oriented E2E-AD baseline under identical baseline conditions.

Besides the late-horizon degradation in L2 and collision rate, MFPAD achieves the lowest average TPC of 1.12 m, the TPC metric further shows that MFPAD produces more temporally consistent trajectories across consecutive replanning steps, which provides additional diagnostic evidence that the proposed design mitigates long-horizon drift rather than only improving pointwise prediction accuracy.

To further examine the stability of the reported gains, we conducted three repeated training runs for MomAD and MFPAD on NuScenes. The averaged results remain consistent with Table II, with limited variation across repeated runs. Specifically, MFPAD achieves 1.19 ± 0.02 m versus 1.41 ± 0.02 m in average L2, 0.80 ± 0.01% versus 0.90 ± 0.02% in average collision rate. A preliminary significance analysis across the three runs yields $p < 0.05$ on the key long-horizon metrics, providing additional statistical support for the consistency of the observed gains.

2) Extended-horizon Analysis Beyond 6 s: To further examine method behavior beyond the primary 6 s setting, we extend the analysis horizon to 12 s and report L2 error and collision rate at 9 s and 12 s, together with the average performance over 7–12 s. As shown in Table III, all methods degrade markedly once the horizon extends beyond 6 s, confirming that longer-horizon planning remains highly challenging. Nevertheless, MFPAD achieves the best performance at both 9 s and 12 s, as well as the best average performance over 7–12 s. These results suggest that, although performance degradation is unavoidable when the horizon is further extended, the proposed memory–forgetting mechanism mitigates long-horizon drift more effectively than the compared baselines.

3) Short-horizon Compatibility (1–3 s): To facilitate comparison with prior planning-oriented E2E-AD methods that mainly report short-horizon performance, and to show that the proposed long-horizon refinement design does not sacrifice near-term planning quality, Table IV summarizes results within the 1–3 s window. MFPAD maintains strong short-horizon performance and achieves the best L2 among the listed methods at 1–3 s, while also reducing the 3 s collision rate from 0.22% [22] to 0.14%. This suggests that the proposed sequence-aware refinement strengthens long-term planning without compromising near-term accuracy.

D. Closed-loop Results on Bench2Drive and NAVSIM

1) Closed-loop validation on Bench2Drive: To further examine whether the open-loop gains of MFPAD transfer to feedback-driven driving performance, we evaluate the proposed head on Bench2Drive under the official closed-loop protocol. The results are summarized in Table V. Compared with the corresponding baselines, MFPAD improves the closed-loop metrics under both settings. Under the MomAD(SD)-based setting, MFPAD improves the Driving Score from 47.91 to

TABLE II

PLANNING PERFORMANCE ON THE NUSCENES VALIDATION SET (1–6 S). LOWER IS BETTER FOR L2, COLLISION, AND TPC, WHILE HIGHER IS BETTER FOR FPS. IN ADDITION TO PLANNING-ORIENTED E2E-AD BASELINES, WE INCLUDE DIFFUSIONDRIVE AND DIVER AS REPRESENTATIVE DIFFUSION-BASED PLANNERS, AND GUIDEFLOW AS A REPRESENTATIVE FLOW-BASED PLANNER, FOR DIRECT LONG-HORIZON COMPARISON. RESULTS FOR MOMAD AND MFPAD ARE ADDITIONALLY REPORTED AS MEAN ± STANDARD DEVIATION OVER THREE REPEATED RUNS

Method	L2 (m) ↓							Collision rate (%) ↓							TPC (m) ↓	FPS ↑
	1s	2s	3s	4s	5s	6s	Avg.	1s	2s	3s	4s	5s	6s	Avg.	Avg.	
UniAD [19]	0.47	0.91	1.35	1.91	2.47	3.07	1.70	0.25	0.36	0.61	0.99	1.64	2.51	1.06	1.58	1.8
SparseDrive [21]	0.43	0.87	1.23	1.75	2.32	2.95	1.59	0.19	0.31	0.56	0.87	1.54	2.33	0.97	1.46	**9.0**
DiffusionDrive [24]	0.35	0.81	0.99	1.65	1.94	2.40	1.36	0.20	0.32	0.45	0.82	1.51	2.23	0.92	1.31	**9.0**
DIVER [25]	0.38	0.75	1.10	1.53	1.98	2.49	1.37	0.13	0.31	0.44	0.80	1.41	2.11	0.87	1.35	6.9
GuideFlow [26]	0.42	0.83	1.21	1.73	2.05	2.63	1.48	0.12	0.22	0.42	0.79	1.44	2.15	0.86	1.34	6.4
SparseDrive + MFPAD head	0.37	0.60	1.01	1.39	2.10	2.82	1.38	0.11	0.21	0.44	0.80	1.49	2.21	0.88	1.32	7.9
MomAD [22]	0.41	0.85	1.13	1.67	1.98	2.45	1.41 ± 0.02	0.17	0.30	0.54	0.83	1.43	2.13	0.90 ± 0.02	1.30	7.8
MFPAD (ours)	**0.33**	**0.57**	**0.89**	**1.29**	**1.76**	**2.30**	**1.19 ± 0.02**	**0.08**	**0.20**	**0.41**	**0.75**	**1.38**	**2.01**	**0.80 ± 0.01**	**1.12**	6.6

TABLE III

EXTENDED-HORIZON ANALYSIS BEYOND THE PRIMARY 6 S SETTING ON THE NUSCENES VALIDATION SET. WE REPORT L2 ERROR AND COLLISION RATE AT 9 S AND 12 S, TOGETHER WITH THE AVERAGE PERFORMANCE OVER 7–12 S. LOWER IS BETTER FOR ALL METRICS

Method	L2 ↓ (m)			Collision ↓ (%)		
	9 s	12 s	Avg.	9 s	12 s	Avg.
SparseDrive [21]	5.21	8.05	5.74	5.73	10.76	6.72
MomAD [22]	4.55	7.45	5.21	5.11	9.49	5.97
MFPAD (ours)	**4.39**	**7.15**	**4.83**	**5.00**	**9.33**	**5.70**

TABLE IV

SHORT-HORIZON PLANNING PERFORMANCE ON NUSCENES (1–3 S). LOWER IS BETTER

Method	L2 (m) ↓			Collision (%) ↓		
	1s	2s	3s	1s	2s	3s
UniAD [19]	0.45	0.70	1.04	0.62	0.58	0.63
VAD [20]	0.41	0.70	1.05	0.03	0.19	0.43
DiffusionDrive [24]	0.27	0.54	0.90	0.03	0.05	0.16
DIVER [25]	-	-	-	0.01	0.05	0.15
GuideFlow [26]	-	-	-	**0.00**	**0.02**	0.18
FocalAD [27]	0.27	0.57	0.96	**0.00**	0.04	0.24
SparseDrive [21]	0.29	0.58	0.96	0.01	0.05	0.18
MomAD [22]	0.31	0.57	0.91	0.01	0.05	0.22
MFPAD (ours)	**0.25**	**0.53**	**0.89**	0.01	0.03	**0.14**

49.43, the Success Rate from 18.11% to 21.08%, and the mean Multi-Ability score from 18.66 to 23.04. Under the SparseDrive-based setting, MFPAD further improves the Driving Score from 44.54 to 50.34, the Success Rate from 16.71% to 22.74%, and the mean Multi-Ability score from 18.85 to 24.86. These results suggest that the proposed memory–forgetting refinement not only improves open-loop trajectory quality, but also leads to stronger downstream driving performance under closed-loop evaluation.

2) Closed-loop comparison on NAVSIM: We further validate MFPAD on the NAVSIM navtest split, and the results are reported in Table VI. MFPAD achieves a PDMS of 91.2, which is higher than all listed baselines, while also reaching 99.2 in NC and 99.0 in DAC. In addition, the proposed method obtains competitive TTC, Comfort, and EP scores, indicating strong overall closed-loop driving quality. These results provide complementary evidence, beyond the NuScenes-style open-loop evaluation, that MFPAD remains effective under simulator feedback and realistic decision-making metrics.

E. Robustness and Ablation Studies

1) Robustness on Adv-NuScenes: We further evaluate robustness on Adv-NuScenes, an adversarially perturbed benchmark designed to elicit unsafe interactions. As reported in Table VII, MFPAD achieves the lowest collision rate at every horizon from 1 s to 6 s. In particular, it reduces the collision rate from 2.40% [22] and 2.43% [21] to 1.70% at 3 s, and further lowers the 6 s collision rate from 3.98% and 4.05% to 2.68%. These results indicate that the proposed memory–forgetting refinement improves safety-related performance under challenging interaction shifts.

2) Ablation Study on Memory and Forgetting: To quantify the contribution of each component, we conduct ablations by removing the LSTM memory branch (*w/o Memory*) or the Transformer forgetting branch (*w/o Forgetting*) while keeping all other settings unchanged. We further include a parameter-matched MLP baseline, a GRU-based head, and a Transformer-only head for controlled architectural comparison.

On NuScenes (Table VIII), the full MFPAD yields the best accuracy and safety at both short and long horizons. In particular, removing either component increases the 6 s L2 error from 2.30 m to 2.59–2.79 m and raises the 6 s collision rate from 2.01% to 3.37%–3.50%, showing that long-horizon stability benefits from both long-term memory and selective forgetting. The forgetting branch adds only modest overhead, reducing the runtime from 7.2 FPS to 6.6 FPS, while improving the 6 s L2 error from 2.59 m to 2.30 m and the 6 s collision rate from 3.37% to 2.01%. Therefore, the proposed forgetting branch should be understood as a targeted accuracy–efficiency trade-off: a limited decrease in runtime is exchanged for a clear gain in long-horizon planning quality and safety-related metrics, especially at 6 s.

We further report model complexity and alternative-head comparisons in Table VIII. Compared with the original MomAD-based setting, MFPAD increases the total parameter count only from 86.484M to 89.380M, while the refinement-head FLOPs increase from 1.505G to 1.549G. Moreover, a parameter-matched MLP baseline, constructed by enlarging the original MomAD planning head while keeping the rest of the framework unchanged, still underperforms MFPAD

TABLE V

OPEN-LOOP AND CLOSED-LOOP COMPARISON ON **BENCH2DRIVE** UNDER THE OFFICIAL EVALUATION PROTOCOL. RESULTS ARE REPORTED UNDER TWO BASE SETTINGS: A MOMAD-BASED SETTING AND A SPARSEDRIVE-BASED SETTING. HIGHER IS BETTER FOR DS, SR, EFFICIENCY, COMFORT, AND MULTI-ABILITY SCORES, WHILE LOWER IS BETTER FOR AVG. L2

Method	Open-loop	Closed-loop				Multi-Ability (%) ↑					
	Avg. L2 ↓	DS ↑	SR (%) ↑	Effi ↑	Comf ↑	Merg.	Overta.	Emerge.	Give Way	Traffic Sign	Mean
UniAD-Base [19]	**0.73**	45.81	16.36	129.21	43.58	14.10	17.78	21.67	10.00	14.21	15.55
VAD [20]	0.91	42.35	15.00	157.94	46.01	8.11	24.44	18.64	**20.00**	19.15	18.07
GenAD [38]	-	44.81	15.90	-	-	-	-	-	-	-	-
MomAD(VAD) [22]	0.87	45.35	17.44	162.09	49.34	9.99	26.31	20.07	**20.00**	20.23	19.32
MomAD(SD) [22]	0.82	47.91	18.11	174.91	51.20	13.21	21.02	18.01	**20.00**	21.07	18.66
MFPAD (Ours)	0.77	49.43	21.08	176.12	52.91	**18.12**	26.74	24.12	**20.00**	**26.21**	23.04
SparseDrive [21]	0.87	44.54	16.71	170.21	48.63	12.18	23.19	17.91	**20.00**	20.98	18.85
MFPAD (Ours)	0.81	**50.34**	**22.74**	**178.21**	**55.89**	16.73	**29.41**	**24.58**	**20.00**	25.44	**24.86**

TABLE VI

CLOSED-LOOP COMPARISON ON THE **NAVSIM** NAVTEST SPLIT. 'C' DENOTES CAMERA-ONLY INPUT AND 'C&L' DENOTES CAMERA-AND-LIDAR INPUT. HIGHER IS BETTER FOR ALL METRICS

Method	Input	NC↑	DAC ↑	TTC↑	Comf.↑	EP↑	PDMS↑
UniAD [19]	C	97.8	91.9	92.9	**100**	78.8	83.4
LAW [39]	C	97.9	92.0	92.9	**100**	79.1	83.5
LTF [40]	C	97.4	92.8	92.4	**100**	79.0	83.8
PARA-Drive [41]	C	97.9	92.4	93.0	99.8	79.3	84.0
DiffRefiner [42]	C	98.4	97.4	95.3	**100**	83.4	89.4
VADv2 [43]	C&L	97.2	89.1	91.6	**100**	76.0	80.9
Hydra-MDP [33]	C&L	98.3	96.0	94.6	**100**	78.7	86.5
TrajHF [44]	C&L	96.3	96.0	91.5	**100**	83.1	86.4
DriveSuprim [32]	C&L	97.8	97.3	93.6	**100**	**86.7**	89.9
DiffusionDrive [24]	C&L	98.2	96.2	94.7	**100**	82.2	88.1
BridgeDrive [31]	C&L	98.2	96.1	94.5	**100**	82.3	88.0
GoalFlow [28]	C&L	98.4	98.3	94.6	**100**	85.0	90.3
DiffE2E [30]	C&L	**99.2**	96.8	**96.7**	**100**	83.6	89.8
DIVER [25]	C&L	98.5	96.5	94.9	**100**	82.6	88.3
MFPAD (Ours)	C&L	**99.2**	**99.0**	95.4	**100**	85.7	**91.2**

TABLE VII

ROBUSTNESS COMPARISON ON ADV-NUSCENES [34]. WE REPORT COLLISION RATE (%, ↓) FOR 1–6 S HORIZONS

Method	Collision Rate (%)↓					
	1s	2s	3s	4s	5s	6s
SparseDrive [21]	**0.02**	0.61	2.43	2.90	3.63	4.05
MomAD [22]	**0.02**	0.58	2.40	2.92	3.55	3.98
MFPAD (ours)	**0.02**	**0.45**	**1.70**	**2.05**	**2.33**	**2.68**

TABLE VIII

ABLATION AND ALTERNATIVE-HEAD COMPARISON ON NUSCENES. WE REPORT L2 ERROR (M, ↓), COLLISION RATE (%, ↓) AT 3 S AND 6 S, TOGETHER WITH FPS (↑), PARAMS (M), AND FLOPS (G). PARAMETER COUNT IS REPORTED FOR THE FULL MODEL, WHILE FLOPS ARE MEASURED AT THE REFINEMENT-HEAD LEVEL

Variant	L2 ↓ (m)		Collision ↓ (%)		FPS ↑	Params (M)	FLOPs (G)
	3s	6s	3s	6s			
w/o Memory	0.99	2.79	0.72	3.50	6.9	86.484	1.505
w/o Forgetting	0.95	2.59	0.70	3.37	7.2	89.380	1.549
Param-matched MLP baseline	1.15	2.87	0.56	2.17	7.4	89.378	1.613
GRU-based head	1.05	2.58	0.53	2.15	6.7	89.182	1.553
Transformer-only head	1.08	2.75	0.59	2.33	7.0	88.654	1.532
MFPAD (ours)	**0.89**	**2.30**	**0.41**	**2.01**	6.6	89.380	1.549

TABLE IX

ABLATION STUDY ON ADV-NUSCENES [34]. METRICS ARE ALIGNED WITH TABLE VIII: L2 ERROR (M, ↓) AND COLLISION RATE (%, ↓) AT 3 S AND 6 S

Variant	L2 ↓ (m)		Collision Rate ↓ (%)	
	3s	6s	3s	6s
w/o Memory	3.00	5.91	2.88	4.50
w/o Forgetting	2.88	5.48	2.80	4.35
Full **MFPAD (ours)**	**2.70**	**4.87**	**1.70**	**2.68**

despite having nearly identical total parameters (89.378M vs. 89.380M) and slightly higher refinement-head FLOPs (1.613G vs. 1.549G). We also compare MFPAD with a GRU-based head and a Transformer-only head under the same 6 s setting. Although both alternatives introduce temporal modeling, they remain inferior to MFPAD on all key long-horizon metrics. These results support that the gain of MFPAD is not explained by model capacity alone, nor by replacing the trajectory head with an arbitrary sequence model, but is related to its explicit memory–forgetting temporal refinement design. We also note that *w/o Forgetting* is a functional ablation rather than a structurally pruned model, and therefore retains the same parameter count and FLOPs as the full MFPAD architecture.

A similar trend is observed on Adv-NuScenes (Table IX). The full model reduces the 6 s collision rate to 2.68%, whereas removing memory or forgetting degrades it to 4.35% and 4.50%, respectively. These results suggest that memory preserves trajectory continuity, whereas selective forgetting is important for correcting stale or noisy cues that accumulate over longer horizons.

3) Dropout Strategy Analysis: To further distinguish the proposed forgetting mechanism from generic regularization, Table X compares different dropout strategies in the forgetting stage, including a no-dropout setting, a decoder feature-dropout baseline, and our memory-token dropout design. As shown in Table X, decoder feature dropout already provides a small improvement over the no-dropout setting, reducing the average L2 error from 1.24 m to 1.22 m and the average collision rate from 0.86% to 0.83%. However, the proposed memory-token dropout achieves the best results, further reducing the average L2 error to 1.19 m and the average collision rate to 0.80%. A similar trend is observed at the 6 s horizon, where the proposed design lowers the L2 error from 2.41 m to 2.30 m and the collision rate from 2.26% to 2.01%. The advan-

TABLE X
COMPARISON OF DIFFERENT DROPOUT STRATEGIES IN THE FORGETTING STAGE ON THE NUSCENES VALIDATION SET. LOWER IS BETTER FOR ALL METRICS

Method	Avg. L2 (m)↓	Avg. Coll. (%)↓	6 s L2 (m)↓	6 s Coll. (%)↓
No dropout	1.24	0.86	2.41	2.26
Decoder feature dropout	1.22	0.83	2.36	2.16
Memory-token dropout (ours)	**1.19**	**0.80**	**2.30**	**2.01**

tage of memory-token dropout over decoder feature dropout suggests that the gain is not merely caused by generic dropout regularization. Instead, perturbing the memory/context tokens before refinement creates an incomplete-memory condition, under which the Transformer decoder and gated fusion module learn to correct unreliable long-horizon cues. Therefore, the forgetting mechanism in MFPAD is better understood as structured memory perturbation followed by adaptive correction, rather than as ordinary feature-level dropout alone.

F. Limitations and Analysis

We discuss the current scope of MFPAD from three aspects. First, regarding scalability to longer horizons, our main setting focuses on 6 s planning, while the extended-horizon results in Table III show that all methods degrade as the horizon is extended beyond 6 s. Nevertheless, MFPAD achieves the best average performance over 7–12 s and exhibits the smallest relative increase from 6 s to 12 s in both L2 error and collision rate. This shows that the proposed memory–forgetting design mitigates long-horizon drift more effectively than the compared baselines. However, the substantial increase at 9 s and 12 s also indicates that trajectory-level refinement alone cannot fully prevent error accumulation under much longer rollouts.

Second, the forgetting mechanism depends on the token dropout probability. In the present work, we adopt a fixed dropout setting that performs well in the evaluated benchmarks, and the ablation results in Table VIII show that removing the forgetting branch leads to clear degradation in both long-horizon L2 error and collision rate. This indicates that selective suppression of stale or noisy temporal cues is important for stable planning. At the same time, we do not perform an exhaustive sweep over the dropout probability itself. This hyperparameter introduces a natural trade-off: too little dropout may leave stale cues insufficiently suppressed, while too much dropout may remove useful motion memory. Therefore, the current results should be interpreted as evidence under the chosen setting ($p_f = 0.2$), rather than as proof of a universally optimal value.

Third, we analyze robustness under challenging conditions. As shown in the Adv-NuScenes results and the closed-loop evaluations on Bench2Drive and NAVSIM, MFPAD improves robustness-related and safety-related metrics under adversarial interactions and feedback-driven driving benchmarks. These results suggest that the proposed refinement head remains effective beyond standard open-loop evaluation. However, this evidence is still limited to the evaluated perturbation models and simulator settings, rather than all extreme distribution shifts or deployment environments.

IV. CONCLUSION

In this work, we study long-horizon trajectory planning in planning-oriented end-to-end autonomous driving. We propose MFPAD, a plug-in refinement head that replaces the conventional MLP trajectory regressor with an LSTM-based memory branch and a Transformer-based forgetting branch with token-level dropout. This sequence-aware design produces temporally consistent, interaction-aware ego trajectories over a 6 s prediction horizon. On NuScenes, MFPAD improves long-horizon performance compared with MomAD, reducing the average L2 error from 1.41 m to 1.19 m and the average collision rate from 0.90% to 0.80%, while maintaining an inference speed of 6.6 FPS. On the adversarial Adv-NuScenes benchmark, MFPAD reduces the 3 s collision rate from 2.40% to 1.70%. In addition, the closed-loop results on Bench2Drive and NAVSIM show that the proposed refinement head also improves downstream driving performance under feedback-driven evaluation. Overall, these results suggest that explicitly modeling long-term memory together with selective forgetting in the planning head is beneficial for improving long-horizon planning quality and yielding more robust behavior under challenging conditions.

REFERENCES

[1] L. Chen, P. Wu, K. Chitta, B. Jaeger, A. Geiger, and H. Li, "End-to-end autonomous driving: Challenges and frontiers," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 46, no. 10, pp. 10 164–10 183, 2024, doi: 10.1109/TPAMI.2024.3435937.

[2] P. S. Chib and P. Singh, "Recent advancements in end-to-end autonomous driving using deep learning: A survey," *IEEE Transactions on Intelligent Vehicles*, vol. 9, no. 1, pp. 103–118, 2024, doi: 10.1109/TIV.2023.3318070.

[3] L. Le Mero, D. Yi, M. Dianati, and A. Mouzakitis, "A survey on imitation learning techniques for end-to-end autonomous vehicles," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 9, pp. 14 128–14 147, 2022, doi: 10.1109/TITS.2022.3144867.

[4] F. Codevilla, M. Müller, A. M. López, V. Koltun, and A. Dosovitskiy, "End-to-end driving via conditional imitation learning," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2018, pp. 4693–4700, doi: 10.1109/ICRA.2018.8460487.

[5] Z. Gao, Y. Mu, C. Chen, J. Duan, P. Luo, Y. Lu, and S. Eben Li, "Enhance sample efficiency and robustness of end-to-end urban autonomous driving via semantic masked world model," *IEEE Transactions on Intelligent Transportation Systems*, vol. 25, no. 10, pp. 13 067–13 079, 2024, doi: 10.1109/TITS.2024.3400227.

[6] C. Sanchez, J. Sanchez, C. Cervantes, R. Ortigoza, V. Guzman, J. Juarez, and M. Aranda, "Trajectory generation for wheeled mobile robots via bézier polynomials," *IEEE Latin America Transactions*, vol. 14, no. 11, pp. 4482–4490, 2016, doi: 10.1109/TLA.2016.7795818.

[7] M. J. Reinoso, L. I. Minchala, P. Ortiz, D. F. Astudillo, and D. Verdugo, "Trajectory tracking of a quadrotor using sliding mode control," *IEEE Latin America Transactions*, vol. 14, no. 5, pp. 2157–2166, 2016, doi: 10.1109/TLA.2016.7530409.

[8] E. Larralde-Ortiz, A. Luviano-Juárez, F. Mirelez-Delgado, and D. Mercado-Ravell, "Donkietown: a low-cost experimental testbed for research on autonomous cars," *IEEE Latin America Transactions*, vol. 21, no. 6, pp. 715–722, 2023, doi: 10.1109/TLA.2023.10172136.

[9] A. Kopacz, E. García González, C. Chira, and J. R. Villar Flecha, "Hybrid adaptive greedy algorithm addressing the multi-robot path planning problem," *IEEE Latin America Transactions*, vol. 23, no. 10, pp. 856–864, 2025, doi: 10.1109/TLA.2025.11150630.

[10] N. S. Monteiro, V. M. Gonçalves, and C. A. Maia, "Motion planning of mobile robots in indoor topological environments using partially observable markov decision process," *IEEE Latin America Transactions*, vol. 19, no. 8, pp. 1315–1324, 2021, doi: 10.1109/TLA.2021.9475862.

[11] M. Mendonça, R. H. C. Palácios, R. Breganon, L. Botoni de Souza, and L. Rodrigues Cintra Moura, "Analysis of the inverse kinematics and trajectory planning applied in a classic collaborative industrial robotic manipulator," *IEEE Latin America Transactions*, vol. 20, no. 3, pp. 363–371, 2022, doi: 10.1109/TLA.2022.9667133.

[12] R. Vieira, E. Argento, and T. Revoredo, "Trajectory planning for car-like robots through curve parametrization and genetic algorithm optimization with applications to autonomous parking," *IEEE Latin America Transactions*, vol. 20, no. 2, pp. 309–316, 2022, doi: 10.1109/TLA.2022.9661471.

[13] F. Ugalde Pereira, P. Medeiros de Assis Brasil, M. A. de Souza Leite Cuadros, A. R. Cukla, P. Drews Junior, and D. F. Tello Gamarra, "Analysis of local trajectory planners for mobile robot with robot operating system," *IEEE Latin America Transactions*, vol. 20, no. 1, pp. 92–99, 2022, doi: 10.1109/TLA.2022.9662177.

[14] W. Zeng, W. Luo, S. Suo, A. Sadat, B. Yang, S. Casas, and R. Urtasun, "End-to-end interpretable neural motion planner," in *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 8652–8661, doi: 10.1109/CVPR.2019.00886.

[15] S. Casas, A. Sadat, and R. Urtasun, "Mp3: A unified model to map, perceive, predict and plan," in *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021, pp. 14 398–14 407, doi: 10.1109/CVPR46437.2021.01417.

[16] A. Sadat, S. Casas, M. Ren, X. Wu, P. Dhawan, and R. Urtasun, "Perceive, predict, and plan: Safe motion planning through interpretable semantic representations," in *Computer Vision – ECCV 2020*, ser. Lecture Notes in Computer Science. Springer International Publishing, 2020, vol. 12348, pp. 414–430, doi: 10.1007/978-3-030-58592-1-25.

[17] P. Hu, A. Huang, J. Dolan, D. Held, and D. Ramanan, "Safe local motion planning with self-supervised freespace forecasting," in *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021, pp. 12 727–12 736, doi: 10.1109/CVPR46437.2021.01254.

[18] 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 *Computer Vision – ECCV 2022*, S. Avidan, G. Brostow, M. Cissé, G. M. Farinella, and T. Hassner, Eds. Cham: Springer Nature Switzerland, 2022, pp. 533–549, doi: 10.1007/978-3-031-19839-7-31.

[19] Y. Hu, J. Yang, L. Chen, K. Li, C. Sima, X. Zhu, S. Chai, S. Du, T. Lin, W. Wang, L. Lu, X. Jia, Q. Liu, J. Dai, Y. Qiao, and H. Li, "Planning-oriented autonomous driving," in *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023, pp. 17 853–17 862, doi: 10.1109/CVPR52729.2023.01712.

[20] B. Jiang, S. Chen, Q. Xu, B. Liao, J. Chen, H. Zhou, Q. Zhang, W. Liu, C. Huang, and X. Wang, "Vad: Vectorized scene representation for efficient autonomous driving," in *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, 2023, pp. 8306–8316, doi: 10.1109/ICCV51070.2023.00766.

[21] W. Sun, X. Lin, Y. Shi, C. Zhang, H. Wu, and S. Zheng, "Sparsedrive: End-to-end autonomous driving via sparse scene representation," in *2025 IEEE International Conference on Robotics and Automation (ICRA)*, 2025, pp. 8795–8801, doi: 10.1109/ICRA55743.2025.11128800.

[22] Z. Song, C. Jia, L. Liu, H. Pan, Y. Zhang, J. Wang, X. Zhang, S. Xu, L. Yang, and Y. Luo, "Don't shake the wheel: Momentum-aware planning in end-to-end autonomous driving," in *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2025, pp. 22 432–22 441, doi: 10.1109/CVPR52734.2025.02089.

[23] H. Caesar, V. Bankiti, A. H. Lang, S. Vora, V. E. Liong, Q. Xu, A. Krishnan, Y. Pan, G. Baldan, and O. Beijbom, "nuscenes: A multimodal dataset for autonomous driving," in *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020, pp. 11 618–11 628, doi: 10.1109/CVPR42600.2020.01164.

[24] B. Liao, S. Chen, H. Yin, B. Jiang, C. Wang, S. Yan, X. Zhang, X. Li, Y. Zhang, Q. Zhang, and X. Wang, "Diffusiondrive: Truncated diffusion model for end-to-end autonomous driving," in *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2025, pp. 12 037–12 047, doi: 10.1109/CVPR52734.2025.01124.

[25] Z. Song, L. Liu, H. Pan, B. Liao, M. Guo, L. Yang, Y. Zhang, S. Xu, C. Jia, and Y. Luo, "Breaking imitation bottlenecks: Reinforced diffusion powers diverse trajectory generation," *arXiv preprint arXiv:2507.04049*, 2025, doi: 10.48550/arXiv.2507.04049.

[26] L. Liu, C. Jia, G. Yu, Z. Song, J. Li, F. Jia, P. Wu, X. Hao, Y. Luo *et al.*, "Guideflow: Constraint-guided flow matching for planning in end-to-end autonomous driving," *arXiv preprint arXiv:2511.18729*, 2025, doi: 10.48550/arXiv.2511.18729.

[27] B. Sun, B. Zhang, J. Lu, X. Feng, J. Shang, R. Cao, M. Zheng, C. Wang, S. Yang, Y. Cao, and Z. Song, "Focalad: Local motion planning for end-to-end autonomous driving," *arXiv preprint arXiv:2506.11419*, 2025, doi: 10.48550/arXiv.2506.11419.

[28] Z. Xing, X. Zhang, Y. Hu, B. Jiang, T. He, Q. Zhang, X. Long, and W. Yin, "Goalflow: Goal-driven flow matching for multi-modal trajectories generation in end-to-end autonomous driving," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2025, pp. 1602–1611, doi: 10.1109/CVPR52734.2025.00157.

[29] D. Dauner, M. Hallgarten, A. Geiger, and K. Chitta, "Parting with misconceptions about learning-based vehicle motion planning," *arXiv preprint arXiv:2306.07962*, 2023, doi: 10.48550/arXiv.2306.07962.

[30] R. Zhao, Y. Fan, Z. Chen, F. Gao, and Z. Gao, "Diffe2e: Rethinking end-to-end driving with a hybrid action diffusion and supervised policy," 2025, doi: 10.48550/arXiv.2505.19516.

[31] S. Liu, W. Chen, W. Li, Z. Wang, L. Yang, J. Huang, Y. Zhang, Z. Huang, Z. Cheng, and H. Yang, "Bridgedrive: Diffusion bridge policy for closed-loop trajectory planning in autonomous driving," in *The Fourteenth International Conference on Learning Representations (ICLR)*, 2026, doi: 10.48550/arXiv.2509.23589.

[32] W. Yao, Z. Li, S. Lan, Z. Wang, X. Sun, J. M. Alvarez, and Z. Wu, "DriveSuprim: Towards Precise Trajectory Selection for End-to-End Planning," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 40, no. 14, 2026, pp. 11 910–11 918, doi: 10.1609/aaai.v40i14.38178.

[33] Z. Li, K. Li, S. Wang, S. Lan, Z. Yu, Y. Ji, Z. Li, Z. Zhu, J. Kautz, Z. Wu, Y.-G. Jiang, and J. M. Alvarez, "Hydra-mdp: End-to-end multimodal planning with multi-target hydra-distillation," *arXiv preprint arXiv:2406.06978*, 2024, doi: 10.48550/arXiv.2406.06978.

[34] Z. Xu, B. Li, H.-a. Gao, M. Gao, Y. Chen, M. Liu, C. Yan, H. Zhao, S. Feng, and H. Zhao, "Challenger: Affordable adversarial driving video generation," *arXiv preprint arXiv:2505.15880*, 2025, doi: 10.48550/arXiv.2505.15880.

[35] Z. Li, Z. Yu, S. Lan, J. Li, J. Kautz, T. Lu, and J. M. Alvarez, "Is ego status all you need for open-loop end-to-end autonomous driving?" in *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024, pp. 14 864–14 873, doi: 10.1109/CVPR52733.2024.01408.

[36] X. Jia, Z. Yang, Q. Li, Z. Zhang, and J. Yan, "Bench2drive: Towards multi-ability benchmarking of closed-loop end-to-end autonomous driving," *Advances in Neural Information Processing Systems*, vol. 37, pp. 819–844, 2024, doi: 10.52202/079017-0025.

[37] D. Dauner, M. Hallgarten, T. Li, X. Weng, Z. Huang, Z. Yang, H. Li, I. Gilitschenski, B. Ivanovic, M. Pavone *et al.*, "Navsim: Data-driven non-reactive autonomous vehicle simulation and benchmarking," *Advances in Neural Information Processing Systems*, vol. 37, pp. 28 706–28 719, 2024, doi: 10.52202/079017-0902.

[38] W. Zheng, R. Song, X. Guo, C. Zhang, and L. Chen, "Genad: Generative end-to-end autonomous driving," in *Computer Vision – ECCV 2024*, A. Leonardis, E. Ricci, S. Roth, O. Russakovsky, T. Sattler, and G. Varol, Eds. Cham: Springer Nature Switzerland, 2025, pp. 87–104, doi: 10.1007/978-3-031-73650-66.

[39] Y. Li, L. Fan, J. He, Y. Wang, Y. Chen, Z. Zhang, and T. Tan, "Enhancing end-to-end autonomous driving with latent world model," in *The Thirteenth International Conference on Learning Representations (ICLR)*, 2025, doi: 10.48550/arXiv.2406.08481.

[40] K. Chitta, A. Prakash, B. Jaeger, Z. Yu, K. Renz, and A. Geiger, "Transfuser: Imitation with transformer-based sensor fusion for autonomous driving," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 45, no. 11, pp. 12 878–12 895, 2023, doi: 10.1109/TPAMI.2022.3200245.

[41] X. Weng, B. Ivanovic, Y. Wang, Y. Wang, and M. Pavone, "PARA-Drive: Parallelized Architecture for Real-Time Autonomous Driving," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Seattle, WA, USA, 2024, pp. 15 449–15 458, doi: 10.1109/CVPR52733.2024.01463.

[42] L. Yin, R. Ju, G. Guo, and E. Cheng, "Diffrefiner: Coarse to fine trajectory planning via diffusion refinement with semantic interaction for end to end autonomous driving," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 40, no. 14, pp. 12 009–12 017, 2026, doi: 10.1609/aaai.v40i14.38189.

[43] S. Chen, B. Jiang, H. Gao, B. Liao, Q. Xu, Q. Zhang, C. Huang, W. Liu, and X. Wang, "Vadv2: End-to-end vectorized autonomous driving via probabilistic planning," *arXiv preprint arXiv:2402.13243*, 2024, doi: 10.48550/arXiv.2402.13243.

[44] D. Li, C. Li, Y. Wang, J. Ren, X. Wen, P. Li, L. Xu, K. Zhan, P. Jia, X. Lang, N. Xu, and H. Zhao, "Learning personalized driving

styles via reinforcement learning from human feedback," *arXiv preprint arXiv:2503.10434*, 2025, doi: 10.48550/arXiv.2503.10434.

Yikai Wu received the B.S. degree from Nanjing University of Science and Technology (China) in 2018. He received a master's degree from Nanjing University of Science and Technology (China) in 2021. He is now a PhD student majoring in Control Science and Engineering at Nanjing University of Science and Technology (China), with a research focus on intelligent transportation systems.

Qizhou Hu received his Ph.D. degree from Southeast University and completed postdoctoral research at Tsinghua University. He has also been a visiting scholar at the University of Hong Kong, the Technical University of Denmark, and the University of Tokyo, Japan. He is currently an Associate Professor and Ph.D. supervisor at Nanjing University of Science and Technology. His research interests include transportation control theory and engineering, systems engineering, control engineering, and artificial intelligence.

Aiguo Lei received the B.S. degree from Henan Polytechnic University (China) in 2019. He received a master's degree from Nanjing University of Science and Technology (China) in 2022. He is now a PhD student in Nanjing University of Science and Technology (China), with a research focus on Intelligent Scheduling for High-Speed Trains and intelligent transportation systems.

Ziying Song received the B.S. degree from Hebei Normal University of Science and Technology (China) in 2019. He received a master's degree from Hebei University of Science and Technology (China) in 2022. He is now a PhD student majoring in Computer Science and Technology at Beijing Jiaotong University (China), with a research focus on Computer Vision.