Expert Systems With Applications 296 (2026) 129191

Contents lists available at ScienceDirect

Expert Systems With Applications

journal homepage: www.elsevier.com/locate/eswa

Driving policy distillation in autonomous racing with adaptive racing vocabulary and optimal driving guidance

Jonghyun Lee [a], Hyunwook Kang [a], Yuseung Na [a], Jeonghun Kang [b], Junhee Lee [b], Seongjae Jeong [a], Jiwon Seok [a], Kichun Jo [a,*]

[a] Hanyang University, 222 Wangsimni-ro, Seongdong-gu, Seoul, 04763, South Korea
[b] Hyundai Motor Group, 12, Heolleung-ro, Seocho-gu, Seoul, 06797, South Korea

ARTICLE INFO

Keywords:
Autonomous driving
Autonomous racing
Racing vocabulary
Bayesian optimization
Mid-to-mid planning

ABSTRACT

Autonomous racing poses unique challenges - including high-speed dynamics, close competition, and operation at the limits of vehicle performance - that are not fully addressed by current learning-based driving policies. In this paper, we propose a specialized neural network driving policy architecture for real-time autonomous racing to tackle these challenges. Our approach introduces an adaptive racing vocabulary that encodes track geometry and vehicle state information, enabling the policy to respond effectively to rapidly changing racing conditions. We further employ policy distillation with multiple cost heads guided by an optimal driving reference, thereby reducing the reliance on large expert-driving datasets. In addition, Bayesian optimization dynamically combines cost components (controllability, safety, speed, etc.), minimizing lap time while maintaining vehicle control. In high-fidelity vehicle dynamics simulations, the proposed architecture demonstrates robust and adaptive driving behavior, successfully handling the complex and demanding scenarios inherent in autonomous racing.

1. Introduction

Autonomous driving research has evolved into two prominent branches: autonomous racing and learning-based driving policy development. Autonomous racing focuses on pushing vehicles around tracks at high speeds without human intervention, thereby presenting extreme conditions and complex interactions that test the limits of autonomous capabilities. Several competitions in this domain have spurred innovation by challenging vehicles to perform in high-speed racing environments (Betz et al., 2023, 2019; Herrmann et al., 2020; Jung et al., 2023; Kabzan et al., 2020; Lee et al., 2024; Na et al., 2024; Ni et al., 2019). In parallel, learning-based driving policy development employs techniques such as imitation learning and reinforcement learning to train neural network models to drive directly from data rather than via handcrafted rules. Recent advances in this area integrate multiple driving tasks within a single neural network, demonstrating the potential for end-to-end learning of robust driving policies (Chen et al., 2024; Hu et al., 2023b; Li et al., 2024a; Weng et al., 2024).

In general driving scenarios, state-of-the-art neural network policies use multi-modal trajectory planning and multi-objective value optimization to handle complex real-world conditions. Multi-modal trajectory planning generates a set of plausible trajectories (a trajectory vocabulary) covering various feasible paths under different conditions (Chen et al., 2024; Li et al., 2024a). This approach provides flexibility in selecting an optimal trajectory and improves interpretability by assigning a value (or probability) to each candidate path; it also naturally allows the candidate set to expand as driving scenarios evolve. In parallel, multi-target value optimization enables policies to balance multiple objectives simultaneously. For example, Hydra-MDP uses multiple value functions to represent different aspects of driving quality (minimizing collisions, staying within drivable areas, maintaining passenger comfort, etc.), and selects the path with the lowest predicted cost among candidates. Similarly, VADv2 incorporates constraints for collision avoidance and lane alignment, ensuring the ego vehicle remains safe and on-course while pursuing its goals. These methods demonstrate the benefit of considering a diverse trajectory set and multiple objectives in general autonomous driving.

Despite advancements in neural network-based driving policies, significant gaps remain in addressing the unique challenges of autonomous racing. Most existing research focuses on general roadway driving and

* Corresponding author.
E-mail addresses: jonghyunlee@hanyang.ac.kr (J. Lee), hyunwookkang@hanyang.ac.kr (H. Kang), yuseungna@hanyang.ac.kr (Y. Na), jeonghun.kang@hyundai.com (J. Kang), junheelee@hyundai.com (J. Lee), sjeong99@hanyang.ac.kr (S. Jeong), jiwonseok@hanyang.ac.kr (J. Seok), kichunjo@hanyang.ac.kr (K. Jo).

https://doi.org/10.1016/j.eswa.2025.129191
Received 1 May 2025; Received in revised form 21 June 2025; Accepted 26 July 2025
Available online 29 July 2025

does not fully account for racing-specific factors such as high-speed dynamics, close competition with other vehicles, and the need to operate at the limits of tire grip. Imitation learning for racing scenarios is further complicated by the difficulty of gathering comprehensive expert datasets and constructing suitable teacher policies that capture the diverse strategies required for competitive racing. Additionally, current methods for generating trajectory vocabularies are inefficient, often relying on general driving data that fail to reflect the distinct geometry and conditions of racing tracks. The process of final trajectory selection typically uses heuristic approaches with static weighting schemes, limiting adaptability to the dynamic nature of racing environments. Finally, common autonomous driving simulators lack the fidelity needed to accurately model critical racing dynamics, hindering both the training and evaluation of NN-based driving policies designed for racing scenarios.

To address the above challenges, we propose a new neural network architecture and training methodology for real-time autonomous racing. The architecture (illustrated in Fig. 1) comprises four main modules: Racing Perception, Racing Policy, Racing Policy Distillation, and Racing Action. **Racing Perception** processes key details of the environment - track geometry and race line, positions of other vehicles, and the ego vehicle's dynamic state - to produce an encoded representation of the racing scenario. **Racing Policy** then generates a set of candidate trajectories (an adaptive trajectory vocabulary) tailored to the current racing conditions, accounting for vehicle dynamics and racing-specific constraints. The environment representation and candidate trajectories are fed into a policy transformer, which uses a multi-layer perceptron (MLP) head to select the most appropriate trajectory for the given scenario. To efficiently evaluate and refine the trajectory selection, we introduce **Racing Policy Distillation**, which estimates multiple cost components for each trajectory using a mathematical optimization-based reference (derived from an offline optimal racing policy). This distilled evaluation provides stable supervisory signals without requiring large expert-driving datasets. Finally, **Racing Action** combines the multiple cost components and dynamically tunes their weights via Bayesian optimization. This ensures that the ultimate trajectory selection is adaptive and optimized to minimize lap time while maintaining safety and control. Our key contributions are as follows:

- We propose a cost-distillation method for racing policies that uses an optimal control solution as a reference to evaluate trajectory quality. This approach provides a reliable supervision signal (pseudo-ground-truth cost) for training the policy, reducing the need for extensive human driving data.
- We introduce an adaptive trajectory vocabulary that incorporates track geometry and ego-vehicle state. This yields a richer set of candidate racing trajectories, improving the coverage of feasible racing maneuvers and enhancing the policy's responsiveness to dynamic changes in the race environment.
- We apply Bayesian optimization to automatically tune the weights used to combine the different trajectory cost components. This weight optimization aims to minimize lap time while maintaining safety, yielding a more balanced and performant driving policy than fixed weighting schemes.
- We validate the proposed architecture in a high-fidelity racing simulator, demonstrating that our learned policy achieves competitive lap times with minimal collisions. The simulator experiments, which accurately model crucial racing dynamics (tire forces, vehicle inertia, etc.), confirm that our approach is effective under realistic racing conditions.

The remainder of this paper is structured as follows. Section 2 provides a brief overview of related research. Section 3 details the method and its components. In Section 4, we present our experimental results, and discussions. Finally, we conclude with Section 5, where we summarize our findings and future research directions.

2. Related work

Autonomous racing. Autonomous racing has established itself as a critical field for pushing the boundaries of autonomous driving technology. The DARPA challenges (Thrun et al., 2006; Urmson et al., 2008) enabled the development of various path-planning and control techniques for general driving scenarios, laying the foundation for further research. In the 2010s, Funke (Funke et al., 2012) proposed a controller that considers tire friction limits to enable stable high-speed driving on a predefined race line. Similarly, Liniger (Liniger et al., 2015) developed a model predictive control-based algorithm to find the shortest path while interacting with multiple competing vehicles. These studies laid the technological groundwork necessary to meet the extreme demands of racing environments. More recently, competitions like Formula Student Driverless (Kabzan et al., 2020; Ni et al., 2019), Roborace (Betz et al., 2019; Herrmann et al., 2020), the Indy Autonomous Challenge (Betz et al., 2023; Jung et al., 2023), and the Hyundai Motor Group Autonomous Driving Challenge (Lee et al., 2024; Na et al., 2024) have fostered research on high-speed driving, extreme driving conditions, and complex vehicle interactions, positioning autonomous racing as an essential testing ground for advancing autonomous driving technologies.

Learning-based driving policy. Recent mid-to-mid based driving approaches have been introduced to improve trajectory selection in high-speed or complex urban driving scenarios, and have significantly contributed to enhancing prediction accuracy. However, most of these methods still rely on L1 loss or L2 loss against ground-truth trajectories, without explicitly considering semantic driving attributes such as safety, traffic rule compliance, and behavioral consistency (Abouelazm et al., 2024). For example, PLUTO (Cheng et al., 2024) improves performance by generating trajectory candidates based on fixed reference lines and distinguishing between static and dynamic agents. PlanScope (Xin et al., 2024) introduces a time-weighted loss function that adjusts the importance of each time step to improve long-horizon prediction. Diffusion Planner (Zheng et al., 2025) leverages a diffusion-based generative model to produce multiple trajectory candidates that account for interactions with surrounding agents. While these methods effectively improve geometric accuracy, they remain limited in terms of semantic reasoning and behaviorally aligned policy learning, which are crucial for robust and interpretable decision-making in high-speed driving.

In parallel, recent end-to-end learning-based driving policies aim to move beyond low-level trajectory imitation and instead focus on learning high-level decision patterns through cost-based optimization. While mid-to-mid methods predominantly rely on imitation learning using supervised regression to ground-truth trajectories, end-to-end approaches formulate the problem as direct policy learning from multi-cost signals, enabling the incorporation of safety, comfort, and rule compliance objectives into training.

Hydra-MDP (Li et al., 2024a) considers collision cost and drivable area compliance cost to ensure the model learns to avoid collisions and stay within drivable areas during driving. This model is structured to select the path with the lowest predicted cost among multiple input trajectory candidates. VAD (Jiang et al., 2023) employs constraints such as ego-agent collision constraint, ego-boundary overstepping constraint, and ego-lane directional constraint to align the ego vehicle's collision and angle with lanes. In case of MP3 (Casas et al., 2021), model learns driving policies by considering comfort and safety costs such as jerk, lateral acceleration, and curvature to improve driving collision and comfort, as well as costs related to following lane centers. Additionally, Dauner et al. (2023), Hu et al. (2023a), Xi et al. (2023) evaluates driving policies using various metrics from the nuPlan dataset (Caesar et al., 2021), such as collision rate, Time-To-Collision (TTC), drivable area compliance, comfort, progress, speed limit, and direction, achieving high performance. In Li et al. (2024b), the need for diversified evaluation metrics is emphasized by proposing the Curb Collision Rate (CCR). CCR evaluates the likelihood of predicted driving paths deviating from

road boundaries, indicating the safety and rationality of driving policies, and demonstrates improved predictive quality when incorporated into training.

While our method adopts a mid-to-mid planning structure, it is inspired by recent end-to-end policies that leverage multi-cost learning to improve behavioral reasoning. In contrast to reinforcement learning (RL)-based approaches such as Wurman et al. (2022), which directly predict control commands from image observations, our imitation learning framework offers several advantages. RL methods, while capable of high performance, often suffer from low sample efficiency, unstable convergence, and high computational cost, making them less practical for real-world deployment in safety-critical domains like autonomous racing.

Building on the above developments, our work targets the open gap between classical autonomous racing approaches and modern learning-based policies. In particular, we propose a learning-driven racing policy that is designed for the extreme conditions of autonomous racing. Our approach synergizes insights from autonomous racing (e.g., explicit handling of vehicle dynamics and race lines) with advanced policy learning techniques (e.g., cost function optimization and neural network flexibility). The aim is to achieve high-speed, safe, and efficient racing performance through end-to-end neural decision-making, something not accomplished by prior methods. This next section details our proposed methodology.

3. Methods

Our proposed system architecture, as shown in Fig. 1, is divided into four main parts. In operation, **Racing Perception** first fuses ego-vehicle data, track geometry (race line and map), and competitor information into a unified state. The **Racing Policy** module then encodes this state and proposes a set of feasible trajectories in real time. These candidate trajectories are passed through a transformer network that uses the environmental encoding as keys/values and the trajectory encodings as queries, outputting feature vectors for each trajectory. **Racing Policy Distillation** evaluates each trajectory's feature vector with several cost heads (e.g., safety, speed, controllability), and these predicted costs are compared to reference optimal costs (computed offline) to compute a training loss. During inference, the cost head outputs serve to score each candidate trajectory. **Racing Action** stage employs Bayesian optimization to determine the optimal cost weights, applying the trained model to select the best trajectory, which is then input into an MPC controller for ego vehicle control.

To complement this high-level description, Algorithm 1 summarizes the entire flow and operational logic during both training and inference stages. The following pseudocode outlines the full pipeline of our proposed Adaptive Racing Policy - from environment encoding and trajectory vocabulary generation to cost-based scoring and final control. It provides a unified view of both the training stage (with cost supervision) and the inference stage (trajectory selection and control execution), mirroring the modular blocks illustrated in Fig. 1.

3.1. Racing perception

3.1.1. Ego state

Ego state refers to the information about the ego vehicle and is essential for ensuring safe and fast racing. Ego state data includes parameters such as position, orientation, speed, and acceleration. In this model, using position, motion, and tire steering angle as inputs demonstrated high performance. This Ego state information can be obtained from the in-vehicle data and GNSS/INS.

3.1.2. Race-line

The race line is the path around the track that a vehicle would ideally follow to minimize lap time. In practice, it balances path length and curvature: a tighter line shortens distance but increases curvature (and thus requires slowing down). Our race-line design is inspired by the minimum-curvature approach of Heilmeier et al. (2020), which emphasizes minimizing curvature to reduce lateral forces on the vehicle. The race line is derived by solving the optimization problem given in Eq. (1).

$$\begin{aligned} &\min_{l_1,\dots,l_N} \sum_{i=1}^{N} \kappa_i^2 \\ &\text{s.t.} \quad l_i = l_N \quad \forall\, 1 \le i \le N \\ &\qquad l_i \in \left[-w^i_{\text{left}} + w_{\text{safe}},\, w^i_{\text{right}} - w_{\text{safe}}\right] \end{aligned} \tag{1}$$

Here N represents the number of points, l_i denotes the lateral deviation of the i-th point from the center line, w_{left} and w_{right} are the widths from the center line to the boundaries at each end of the track, and w_{safety} is a safety margin that accounts for the vehicle's width. Each point on the race line possesses the following attributes: $X, Y, \psi, \kappa, v_x, a_x$.

3.1.3. Race track map

The race track map represents the drivable surface within the curbs of the racing track, indicating the area where the vehicle can operate. If the vehicle moves beyond this area, the changing road surface conditions can make it difficult to maintain maximum speed, significantly

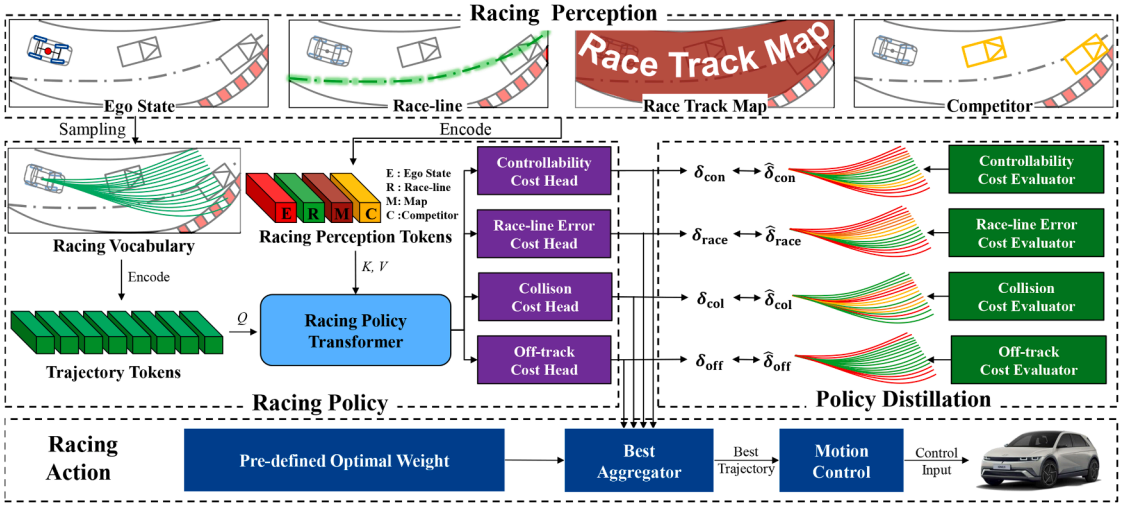

Fig. 1. System architecture. Driving policy distillation in autonomous racing system architecture with adaptive racing vocabulary and optimal driving guidance.

Algorithm 1 Adaptive racing policy with cost distillation.

Require: Ego state E, Race-line R, Track map M, Competitor info C
Ensure: Control command u
// **Racing perception and environment encoding**
Encode E, R, M, C individually
$E_{\text{env}} \leftarrow \text{SelfAttention}(\text{Concat}(\text{Enc}(E), \text{Enc}(R), \text{Enc}(M), \text{Enc}(C)) + \text{PE})$
▷ E_{env}: used as Key/Value
// **Adaptive racing vocabulary generation**
$V \leftarrow \emptyset$ ▷ Initialize planning vocabulary set
for all acceleration $a \in [a_{\min}, a_{\max}]$ **do**
  $S \leftarrow \text{GenerateLongitudinalProfile}(a)$
  for all lateral offset $n \in [n_{\min}, n_{\max}]$ **do**
    $N \leftarrow \text{GenerateLateralProfile}(n)$
    $\tau \leftarrow \text{TransformFrenetToCartesian}(S, N, R)$
    $V \leftarrow V \cup \{\tau\}$
  end for
end for
$V' \leftarrow \text{Encode}(V) + \text{PE}$ ▷ Planning vocabulary embedding
// **Transformer decoder with QKV attention**
$V'' \leftarrow \text{TransformerDecoder}(Q = V', K, V = E_{\text{env}})$
// **Cost prediction using scoring heads**
for all cost head $i \in \{\text{lat}, \text{lon}, \text{col}, \text{off}, \text{stab}, \text{trace}\}$ **do**
  $\delta_i \leftarrow \text{MLP}_i(V'')$
end for
// **Cost supervision via optimal policy (Training only)**
for all trajectory $\tau \in V$ **do**
  for all cost i **do**
    $\hat{\delta}_i \leftarrow \text{EvaluateOptimalCost}_i(\tau)$
    $L_i \leftarrow \text{MSE}(\delta_i, \hat{\delta}_i)$
  end for
end for
$L_{\text{total}} \leftarrow \sum_i L_i$
// **Offline Bayesian optimization for cost weights**
$\{w_i\} \leftarrow \text{BayesianOptimization}\left(\sum_i w_i \cdot \hat{\delta}_i \rightarrow \min \text{racing time}\right)$ ▷ Performed offline using optimal cost labels before training/inference
// **Trajectory selection at inference**
$\text{AggregateCost} \leftarrow \sum_i w_i \cdot \delta_i$
$\tau_{\text{best}} \leftarrow \arg\min_\tau(\text{AggregateCost})$
// **Vehicle motion control**
$u \leftarrow \text{MPC}(\tau_{\text{best}}) + \text{PID}(\tau_{\text{best}})$
return u

affecting control and stability. In this model, the race track map is represented using the center line and width information of the track boundaries. The race track map can be obtained through real-time inference, such as semantic segmentation using camera or LiDAR sensors, or by utilizing positioning systems to retrieve information from high-definition map.

The track map defines the drivable area of the circuit, typically bounded by curbs or walls. Going outside this area drastically reduces tire grip and can cause loss of control. We represent the track map by the centerline geometry plus the track half-width at each point. This representation allows the system to know how far left or right it can deviate from the centerline at any given segment. The necessary map information can be obtained from a prior high-definition map of the track or through on-board perception. In our implementation, we assume the track boundaries are known and provide the model with the centerline and boundary offsets. This track map knowledge ensures that any planned trajectory stays within legal boundaries, which is critical for safety.

3.1.4. Competitor

Competitor refers to other vehicles on the racing track that compete with the ego vehicle. In this model, the past 1-s positions and speeds of Competitors are utilized at 0.1-s intervals, with a padding mask used at each point to determine the presence of competitor tracking, ensuring the model does not focus on irrelevant information. Competitor's information can be obtained through real-time detection and tracking using camera and LiDAR sensors.

The presence and behavior of other vehicles on track must be perceived to avoid collisions and to plan overtaking maneuvers. We encode competitor information by recording each detected opponent's recent trajectory and speed. In our setup, we sample each competitor's position and velocity over the past 1 s at 0.1 s intervals. This yields a sequence of up to 10 positions for each detected competitor. A binary mask at each timestep indicates whether a valid competitor observation is present, preventing the model from treating absent vehicles as real obstacles. This sequence representation allows the policy to infer both current positions and short-term motion trends of opponents. Competitor data is obtained via real-time detection and tracking algorithm. Using this information, the Racing Perception module produces an encoded competitor context that will inform the policy about nearby vehicles and their trajectories.

3.2. Racing policy

3.2.1. Environment encoder

To account for the interactions between input data in the Racing Perception module, the Environment Encoder encodes each input individually and then integrates them using self-attention. This process generates features that reflect the surrounding environmental information, which are used as the key and value for the Racing Policy Transformer Decoder to select the optimal racing vocabulary. The equation for the Environment Encoder can be found in Eq. (2).

$$E_{\text{env}} = \text{SA}(\text{Concat}(\text{En(M)}, \text{En(E)}, \text{En(R)}, \text{En(C)}) + \text{PE}) \tag{2}$$

where SA denotes self-attention, En represents encoding, and PE stands for positional encoding, while M, E, R, and C refer to the race track map, ego-state, race line, and competitor, respectively.

The features generated by the Environment Encoder, which take input data such as Ego state and competitor information processed through pre-processing and specific algorithms, can be replaced with BEV (Bird's Eye View) features directly extracted from the camera raw sensor data. This demonstrates that the proposed model structure offers flexibility and scalability for easy extension to an end-to-end autonomous driving model.

3.2.2. Racing vocabulary

The racing vocabulary is constructed in two stages, ensuring that the trajectories reflect the optimal path along the race line while accommodating road geometry variations in Frenet frame inspired by Werling et al. (2010). First, longitudinal profiles are generated using a Constant Acceleration (CA) model by sampling accelerations from maximum deceleration to maximum acceleration over a fixed time horizon T. Each profile computes positions s_t at time step t using:

$$s_t = s_{t-1} + v_{t-1} \cdot \Delta t + \frac{1}{2} a_{t-1} \cdot \Delta t^2, \tag{3}$$

where v_{t-1} and a_{t-1} are the velocity and acceleration at the previous step. This method produces a set of longitudinal vocabularies covering a range of acceleration profiles.

Second, a lateral profile is generated for each longitudinal trajectory using lateral deviations n from the race line. Starting from the vehicle's current lateral position n_0 and transitioning to a target position n_f, a 5th-order polynomial ensures smooth lateral movement. The lateral trajectory n_t at time step t is defined as:

$$n_t = a_0 + a_1 t + a_2 t^2 + a_3 t^3 + a_4 t^4 + a_5 t^5 \tag{4}$$

where the coefficients a_0, a_1, a_2, a_3, a_4 and a_5 are determined based on boundary conditions for lateral position, velocity, and acceleration. After then, these longitudinal and lateral profiles (s_t, n_t) are combined and transformed from the Frenet to the Cartesian frame.

The racing vocabulary encoder, represented as En in Eq. (5), encodes input paths using 1D convolution and global feature aggregation to generate high-dimensional embedding features that combine both local and global characteristics of the input racing vocabulary. Subsequently, PE (positional encoding) values are added to each query, following the approach described in Chen et al. (2024).

$$E_{\text{query}} = \text{En}(\text{RacingVocab}) + \text{PE} \tag{5}$$

3.2.3. Racing policy transformer decoder

The Racing Policy Transformer Decoder module learns the interaction between the encoded racing vocabulary query and the racing environment key-value information through an attention-based transformer, applying self-attention, multi-head cross-attention, and feed-forward layers with normalization and dropout to refine and transform the input query features, as shown in Eq. (6).

$$E_{\text{feat}} = \text{Transformer}((Q = E_{\text{query}}, K, V = E_{\text{env}}) \tag{6}$$

The decoder outputs feature vectors that feed the individual cost-scoring heads, enabling each head to predict its corresponding cost component.

3.2.4. Cost scoring heads

This module uses a simple multi-layer perceptron (MLP) structure to process the input vocabulary decoder query vectors. Each query is passed through linear layers with ReLU activation, ultimately mapping to a size of queries to generate predictions for each racing vocabulary trajectory. This process is repeated for each cost decoder, resulting in predictions for all racing vocabulary paths for each cost decoder as shown in Eq. (7).

$$\delta_i = \text{MLP}_i(E_{\text{feat}}), \tag{7}$$

where i denotes cost components, which broadly include four main costs: racing controllability cost, race line error cost, collision cost, and off-track cost. More specifically, it consists of six detailed costs: lateral distance, lateral acceleration, speed error, collision, drivable area, and controllability. Each MLP head is trained to regress its associated cost term. These cost elements will be discussed in detail in Section 3.3.

3.3. Racing policy distillation

3.3.1. Optimal cost evaluator

To distill the racing policy, we developed four types of cost evaluators and applied them to calculate the cost for each path in the racing vocabulary. The cost evaluators used are as follows.

Racing controllability cost evaluator. Each racing vocabulary is generated laterally in a geometric manner and longitudinally in a kinematic manner, which may lead to uncertainties in real-world control effectiveness. To address this, a racing controllability cost evaluator uses predictive control on a dynamic vehicle model to assess tracking performance and control stability in advance.

As illustrated in Fig. 2, we use a dynamic nonlinear bicycle model that represents the vehicle as a rigid body with mass m and moment of inertia I_z. The state vector is defined as $\mathbf{x} = [x, y, \psi, v_{\text{lon}}, v_{\text{lat}}, \gamma]^T$, where x, y, ψ are the vehicle's global position and orientation, $v_{\text{lon}}, v_{\text{lat}}$ the longitudinal and lateral velocities in the vehicle frame, and γ the yaw rate. The control input vector is $\mathbf{u} = [\delta_f, F_{\text{driving}}]$, with δ_f as the front wheel steering angle and F_{driving} the driving force. The model's dynamics are given by:

$$\dot{\mathbf{x}} = \begin{bmatrix} v_{\text{lon}}\cos(\psi) - v_{\text{lat}}\sin(\psi) \\ v_{\text{lon}}\sin(\psi) + v_{\text{lat}}\cos(\psi) \\ \gamma \\ \frac{1}{m}\left(F_{x_r} - F_{y_f}\sin(\delta_f) + m v_{\text{lat}}\dot{\psi}\right) \\ \frac{1}{m}\left(F_{y_r} + F_{y_f}\cos(\delta_f) - m v_{\text{lon}}\dot{\psi}\right) \\ \frac{1}{I_z}\left(l_f\left(F_{y_f}\cos(\delta_f)\right) - l_r F_{y_r}\right) \end{bmatrix} \tag{8}$$

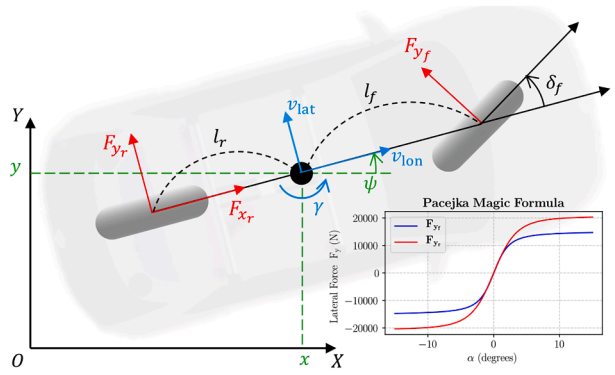

Fig. 2. Dynamic bicycle model. Schematic drawing of dynamic bicycle model and lateral tire forces based on the Pacejka Magic Formula. If the slip angle α exceeds the limit, the tire begins to slide laterally.

Assuming the vehicle is rear-wheel drive, the longitudinal force $F_{x_r} = F_{\text{driving}} - 0.5\rho A_c C_d v_{\text{lon}}^2$ is simply modeled considering aero drag where ρ, A_c and C_d represent air density, frontal area of the vehicle and drag coefficient, respectively. The lateral tire forces $F_{y_{f,r}}$ where the tire forces are represented by the Pacejka Magic Formula are modeled as follows:

$$\begin{aligned} F_{y_{f,r}} = & D_{f,r}\sin\left(C_{f,r}\arctan\left(B_{f,r}\alpha_{f,r}\right.\right. \\ & \left.\left. - E_{f,r}\left(B_{f,r}\alpha_{f,r} - \arctan(B_{f,r}\alpha_{f,r})\right)\right)\right). \end{aligned} \tag{9}$$

where the side slip angles α_f and α_r are given by:

$$\begin{aligned} \alpha_f &= \delta_f - \arctan\left(\frac{v_{\text{lat}} + l_f \cdot \dot{\psi}}{v_{\text{lon}}}\right), \\ \alpha_r &= \arctan\left(\frac{l_r \cdot \dot{\psi} - v_{\text{lat}}}{v_{\text{lon}}}\right) \end{aligned} \tag{10}$$

By simulating each trajectory with this bicycle model, we can accurately predict lateral slip and other dynamic effects. Accordingly, we apply a simple lateral and longitudinal controller to this model, utilizing the Stanley method (Thrun et al., 2006) for lateral control and a PID controller (Na et al., 2024) for longitudinal control, to simulate vehicle behavior when tracking the specified racing vocabulary. The controllability is then evaluated for each i-th racing vocabulary using the following two costs:

$$\begin{aligned} \hat{\delta}_{\text{con},i}^{\text{stability}} &= \sum_{k=1}^{N-2}\left(|j_{\text{lat},k}| + |j_{\text{lon},k}|\right), \\ \hat{\delta}_{\text{con},i}^{\text{traceability}} &= \sum_{k=1}^{N}\|\mathbf{p}_k^{\text{voca}} - \mathbf{p}_k^{\text{pred}}\|_2 \end{aligned} \tag{11}$$

The stability cost $\hat{\delta}_{\text{con}}^{\text{stability}}$ for the racing vocabulary evaluates stability by calculating the jerk values in both the lateral and longitudinal directions. This is achieved by taking the second derivative of each direction's velocity over N predictions. The traceability cost $\hat{\delta}_{\text{con}}^{\text{traceability}}$ assesses tracking accuracy by calculating the positional error between the vocabulary trajectory and the prediction. Here, $\mathbf{p}^{\text{voca}}$ and $\mathbf{p}^{\text{pred}}$ represent the position vectors of the racing vocabulary and the prediction, respectively, and the traceability cost is defined as the sum of Euclidean distances between corresponding points on these paths.

Race-line error cost evaluator. The race line cost evaluator assesses how well the sampled racing vocabulary, generated from racing perception input, follows the given race line. In the absence of competitors, adherence to the race line is crucial for minimizing lap times, making race line tracking a vital factor for optimal racing performance. The cost is calculated based on the longitudinal and lateral tracking errors with respect to the offline-generated race line.

$$\hat{\delta}_{\text{race},i}^{\text{lat}} = \frac{\sum_{k=1}^{N}|n_k|}{N} \tag{12}$$

$$\hat{\delta}^{\text{lon}}_{\text{race},i} = \frac{\sum_{k=1}^{N} |e_{v,k}|}{N}, e_{v,k} = v_k - v_{\text{race},k} \tag{13}$$

As shown in Eq. (12), the lateral error $\hat{\delta}^{\text{lat}}_{\text{race},i}$ is evaluated by the mean absolute lateral deviation between each point of a specific vocabulary and the race line path. The lateral deviation is denoted as n_k at k-th point in i-th racing vocabulary containing a total of N points. For the longitudinal error, as depicted in Eq. (13), the cost is determined by the mean speed error $e_{v,k}$ between the speed profile v_k of the racing vocabulary and that of the race line $v_{\text{race},k}$.

Collision cost evaluator. The collision cost evaluator assesses potential collisions between sampled racing vocabulary and competitors over a defined time horizon and imposes penalties. Since multiple vehicles share the track, calculating the potential collisions is essential to mitigate the risk. The cost is calculated using the first time at which sampled vocabulary and competitors collide. Vehicles are approximated as three circles and the distance between circles are used to calculate the collision at each time step. Compared to the previous methods that merely detect the occurrence of a collision, we use the time of collision as the penalty. It enables comparison by differentiating penalties based on collision timing, assigning larger penalties for imminent collisions and smaller penalties for those occurring later. The collision cost for each i-th racing vocabulary is:

$$\hat{\delta}_{\text{col},i} = -min(t_{\text{collision}}, t_{\text{horizon}}) \tag{14}$$

where $t_{\text{collision}}$ is the first time the collision occurred and t_{horizon} is the time horizon.

Off-track cost evaluator. The off-track cost evaluator assesses the time by which a sample departs the track boundaries. Since samples that exceed these boundaries increase driving risk and lap times, accounting for this cost is essential. The cost is calculated by comparing the sample vocabulary with the track boundary. Similar to the collision cost evaluator, the time at which the point in sample vocabulary first departs the track is used as the basis for cost calculation, with larger penalties assigned to vocabulary that departs the track earlier and smaller penalties for those who depart the track later. This allows for comparative evaluation among track departure vocabularies. The off-track cost for each i-th racing vocabulary is:

$$\hat{\delta}_{\text{off},i} = -min(t_{\text{off}}, t_{\text{horizon}}) \tag{15}$$

where t_{off} is the first time the point in sample vocabulary depart the track.

3.3.2. Training of cost scoring heads

To calculate the loss between the predicted results of each cost in the racing policy stage and the ground truth values extracted from the policy distillation stage, we used MSE loss for each cost decoder, as shown in Eq. (16).

$$L_i = \text{MSE}\left(\delta_i, \hat{\delta}_i\right), i \in \{\text{stability, traceability, lat, lon, col, off}\}, \tag{16}$$

After calculating the MSE loss for each cost decoder, we summed all the loss values, as shown in Eq. (17), to train the model in a way that effectively reflects the racing policy distillation. Furthermore, we confirmed that no significant gradient interference occurred between individual cost decoders and the aggregated cost model during training.

$$L_{\text{total}} = \sum_{i \in \{\text{stability,traceability,lat,lon,col,off}\}} L_i. \tag{17}$$

3.4. Racing action

3.4.1. Best trajectory aggregator

To combine the costs evaluated by the Optimal Cost Evaluator effectively, we used a two-step process involving normalization and optimization. First, each cost metric was standardized using Z-score normalization to ensure comparability despite differing scales and units; mean and standard deviation values were computed from a variety of racing scenarios, and each cost was adjusted accordingly. Second, Bayesian optimization was applied to determine optimal weights for these normalized costs, treating each weight as a hyperparameter that influences overall performance. The resulting weights assigned to the cost terms were as follows: a weight of 0.88 for lateral distance, 0.03 for lateral acceleration, 1.55 for speed error, 0.94 for collision, 0.62 for stability, 0.90 for controllability, and 0.96 for drivable-area adherence. This optimization aimed to balance the primary racing objectives minimizing lap time while avoiding collisions by dynamically prioritizing speed and safety. As a result, the aggregated cost metric became more accurate and effective for autonomous racing scenarios.

3.4.2. Vehicle motion control

Vehicle motion control calculates appropriate steering angles, accelerator pedal, and brake pedal values to ensure accurate and stable tracking of the trajectory selected by the Best Aggregator. By separating the given trajectory into a geometric reference path and a target speed profile, and applying a decoupled lateral and longitudinal controller for each, precise tracking performance can be achieved in both directions. Accordingly, following the control methodology proposed by Na et al. (2024), we implemented an MPC with an understeer compensator to account for tire slip in high-speed racing conditions, providing accurate and stable steering output. Additionally, we employed a PID controller in conjunction with a wheel torque map to achieve precise longitudinal speed tracking.

4. Experiments

4.1. Experimental settings

We trained the dataset collected through rule-based cost selection driving using a single RTX 3090 GPU. During model training, the batch size was fixed at 128, and each model was trained for 100 epochs. The initial learning rate was set to 0.001, and the weight decay was set to 1×10^{-4}. The total training time per model was approximately 2 h.

The racing environment to be validated was created within the CarMaker simulator, replicating the Yongin Speedway, a real-world track located in South Korea. To simulate a real racing scenario, a total of 12 vehicles, including the ego vehicle, were generated for the test, with the ego vehicle starting from the last grid position to encounter various driving situations such as overtaking and adaptive cruise control. Additionally, the maximum speed of surrounding vehicles was varied between 90 and 125 km/h to create conditions similar to an actual racing environment. The simulation was executed at 1000 Hz. All simulations were conducted on a workstation equipped with an Intel(R) Core(TM) i7-14700KF, 64 GB RAM, and an NVIDIA RTX 4080 GPU.

4.2. Metrics

Metrics. To evaluate the effectiveness of the proposed structure, we used three evaluation metrics: **racing time**, **collision count**, and **inference time**, aiming to identify a model that provides fast, safe, and real-time driving capabilities. Racing time represents the time it takes for the ego vehicle to complete one lap around the track, while collision count indicates the number of collisions with competitors during one lap. The collision occurrences were measured using the collision sensor in the CarMaker simulator. Additionally, inference time was used as a critical metric to assess whether the model can operate in real-time within the racing environment. All metrics were measured as the average values over 10 laps around the Yong-in Speedway.

4.3. Racing policy distillation

To verify the effectiveness of the racing policy distillation method based on six racing-related costs, we compared its performance with

Table 1
Performance of the racing-vocabulary generation methods. "DNF" (Did Not Finish) indicates that the vehicle left the track and failed to complete the 10-lap test.

Teacher/Student	Policy method	Racing time (s)	Collision count	Inference time (ms)
Teacher	Optimal Driving Guidance	120.30	0.461	60.00
Teacher	Expert Driver	123.73	0.35	–
Student	Expert Driver	– (DNF)	– (DNF)	30.14
Student	Optimal + Expert Driver	123.61	0.143	30.14
Student	Optimal Driving Guidance	**119.25**	**0.1**	**30.14**

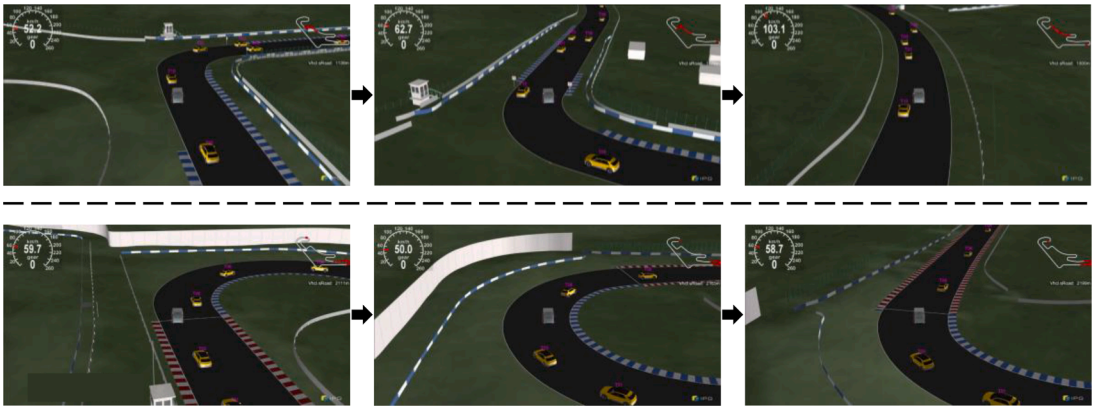

Fig. 3. Qualitative examples of the multi-cost-distilled policy. Top row describes the ego vehicle entering a sharp 90 ° right-hand corner, identifying a slower car ahead, and completing a clean inside-line overtake while remaining within track limits. Bottom row describes a high-curvature U-turn in which the policy keeps a tight line at the tyre-grip limit and exits the corner in a stable posture. These images illustrate how the multi-cost-distilled policy achieves fast lap times while still enabling confident overtaking and reliable cornering in demanding race scenarios.

an imitation learning approach that mimics expert driving data. The same driving scenario environment and racing data generation method were applied to validate the impact of the multi-cost-based racing policy distillation. Additionally, for a comprehensive performance evaluation, comparisons were conducted with the method used during training and the expert driving method. The imitation learning approach imitating expert driving data minimized the expert driving cost by calculating the L2 loss between the actual driving trajectory and the predicted trajectory, thereby performing policy distillation. As shown in the first row of Table 1, the model that mimics expert driving went off the racing track during the drive and failed to complete the lap. In contrast, the results of training using the multi-cost-based policy distillation, as presented in the third row of Table 1, showed a racing time of 119.25 s and 0.1 collisions per laps. In case of combining the expert driving cost with the six driving policy costs for training and driving, the model achieved a racing time of 123.61 s and a collision count of 0.143 per lap. This demonstrates that the multi cost-based policy distillation outperformed the expert imitation-based approach. In racing scenarios, where vehicles operate near their dynamic limits, maintaining control is important, as events such as vehicle spin or unstable cornering may negatively affect lap time and collision risk. To address this, the proposed model incorporates dedicated MLP heads that predict stability and traceability costs. No instances of uncontrollable behavior were observed during testing, indicating that the model achieved both aggressive and stable driving performance within the physical constraints of the vehicle. As a qualitative complement to the numerical results in Table 1, Fig. 3 highlights two representative maneuvers executed by the multi-cost-distilled policy. The first sequence (top row) captures the ego vehicle in an acute, 90-degree right-hand corner: immediately after turn-in, the policy identifies a slower lead car, finely modulates throttle, and completes a clean overtake on the inside line while remaining within track limits. Taken together, Fig. 3 support the quantitative results, showing that the proposed policy delivers competitive lap times, performs clean overtakes, and maintains stable cornering even in challenging racing conditions.

Table 2
Performance comparison according to racing vocabulary generation method.

Racing vocabulary	Racing time (s)	Collision count
Open Dataset-based	– (DNF)	– (DNF)
Expert Driver-based	125.82	1.38
Adaptive Generation-based	**119.25**	**0.1**

4.4. Racing vocabulary generation method

To validate the effectiveness of the racing vocabulary generation method in real-time, which dynamically adapts to the ego vehicle and surrounding environment, we conducted comparisons with various existing trajectory vocabularies. For this purpose, we compared the performance of the proposed model using trajectory vocabularies generated from an open dataset (Wilson et al., 2023) and those derived from expert driver trajectories. To maintain consistency in the evaluation, the number of trajectories was fixed at 256 for all cases, with the open dataset and expert driver trajectories clustered into 256 representative trajectories using k-means clustering.

As shown in Fig. 4, the diverse driving trajectories of the open dataset are well reflected in the trajectory vocabulary. However, due to the domain gap between the urban environment where the open dataset (Wilson et al., 2023) was collected and the racing environment used for validation, it could not accurately represent the racing driving conditions. Consequently, this resulted in DNF with going off the racing track. On the other hand, Table 2 illustrates the result of extracting 256 paths from the expert driver's trajectory vocabulary using k-means clustering. Since this data was collected in the same domain as the actual racing validation environment by expert drivers, it demonstrated faster racing times and a lower collision count compared to the trajectory vocabulary based on the open dataset (Wilson et al., 2023).

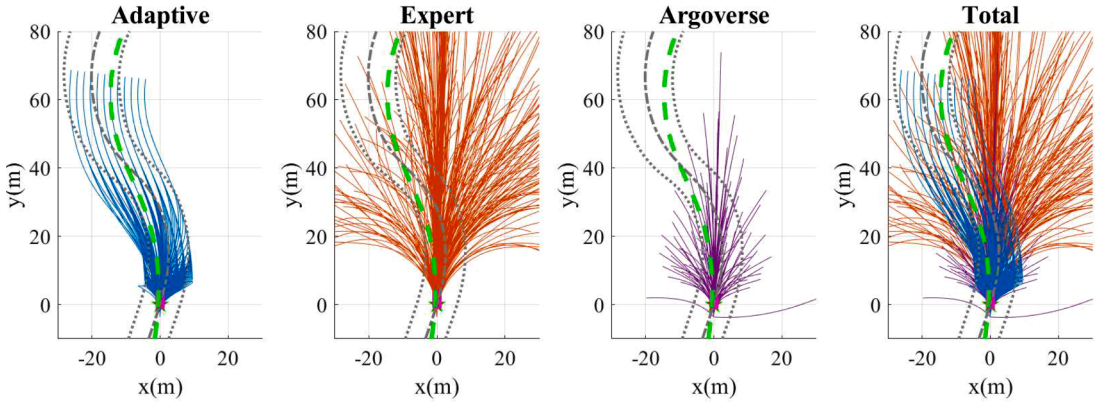

Fig. 4. Comparative analysis of racing vocabulary. Adaptive generation method creates a vocabulary tailored to the racing perception environment, while the expert vocabulary is based on trajectories driven on the same racing track, and the open dataset (Wilson et al., 2023) vocabulary is derived from different environments. Figure on the right visualizes the combination of these three vocabularies.

Table 3
Performance comparison according to number of racing vocabulary.

Racing vocabulary number	Racing time (s)	Collision count	Inference time (ms)
32	132.5348	2.375	**15.96**
256	**119.25**	**0.1**	30.14
4096	– (DNF)	– (DNF)	2349

However, when compared with the Adaptive Trajectory Vocabulary Generation method, which generates optimized trajectories for dynamically changing driving environments, the expert driver's trajectory vocabulary showed lower performance in terms of racing time and collision count. This demonstrates that our proposed racing vocabulary generation method offers more flexible and context-appropriate trajectories.

4.5. Number of racing vocabulary

We conducted experiments to determine the optimal number of racing vocabulary trajectories for the generation method by analyzing metric results for vocabulary sizes of 32, 256, and 4096 trajectories. The results, summarized in Table 3, show that when generating 32 trajectories, the inference time was the shortest due to the fewer number of trajectory paths. However, the likelihood of including a fast and safe trajectories that fully includes the surrounding environment in the vocabulary was reduced, leading to longer racing times and the highest collision count. Conversely, generating 4096 trajectories led to a long computation time due to the pre-processing, embedding, and encoding of a large number of trajectories, making it impossible for the model to follow the predicted trajectory in real-time. When 256 trajectories were generated, the trade-off between safe and fast driving and fast inference speed for real-time racing was balanced most effectively, yielding a racing time of 119.25 s with 0.1 collisions, and an inference time of 30.14ms.

4.6. Bayesian optimization-based weight determinant for min-time racing

By comparing the results of cost weights determined through Bayesian Optimization with those where all cost weights were set to 1, we validated the effectiveness of the Bayesian Optimization-based weight determination method in terms of min-time racing. The weight values derived for the racing scenario through Bayesian Optimization are presented in Table 4. In terms of racing time and collision count metrics, the Bayesian-based weight determination method achieved a racing time of 119.25 s with 0.1 collisions, outperforming the results of the uniform weight setting, which showed 198.21 s with 4.37 collisions.

Table 4
Performance comparison according to cost weight distribution method. $W_{Optimal}{}^{*}$: Result of bayesian optimization, $W_{Init}{}^{*}$: All weights are equal to 1.0.

	Racing time (s)	Collision count
$W_{Optimal}{}^{*}$	**119.25**	**0.1**
$W_{Init}{}^{*}$	198.21	4.37

5. Conclusion

We proposes a driving policy distillation method for autonomous racing that integrates an adaptive racing vocabulary and optimal driving guidance. The method introduces an optimal cost evaluator and cost scoring heads to evaluate candidate trajectories and select the optimal trajectory through policy distillation. The validity of the proposed approach was verified through experiments using various racing vocabularies. Experimental results in a high-fidelity racing simulator showed that the proposed method achieved an average of 0.1 collisions per lap and a lap time of 119.25 s. Compared to fixed vocabularies generated from expert and public datasets, the adaptive vocabulary generation approach reduced lap time by at least 6 s and decreased the average number of collisions by more than one. Bayesian optimization was applied to dynamically adjust cost weights, reducing lap time from 198.21 s to 119.25 s. A setting that generated 256 trajectories balanced performance and inference speed best, with a single inference time of 30.14 ms. Future work will focus on extending the method into an end-to-end policy learning framework using raw sensor inputs to replace manual perception modules and build a unified network that learns features directly from sensor data. Real-vehicle experiments will be conducted to bridge the simulation-to-real gap and evaluate robustness in various driving environments.

CRediT authorship contribution statement

Jonghyun Lee: Writing – original draft, Writing – review & editing, Methodology; **Hyunwook Kang:** Methodology, Investigation; **Yuseung Na:** Methodology; **Jeonghun Kang:** Data curation; **Junhee Lee:** Formal analysis; **Seongjae Jeong:** Validation; **Jiwon Seok:** Data curation; **Kichun Jo:** Conceptualization, Supervision, Project administration.

Data availability

Data will be made available on request.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

This paper was supported by the National Research Foundation of Korea (NRF) grant funded by the Korean Government Ministry of Science and ICT (MSIT) under grant No. RS-2023-00209252 and the Technology Innovation Program (Open-source Based Open Multi SW Platform Technology) funded by the Ministry of Trade Industry and Energy (MOTIE), South Korea, under grant RS-2024-00407710.

Supplementary material

Supplementary material associated with this article can be found in the online version at 10.1016/j.eswa.2025.129191.

References

Abouelazm, A., Michel, J., & Zöllner, J. M. (2024). A review of reward functions for reinforcement learning in the context of autonomous driving. In *2024 IEEE Intelligent vehicles symposium (IV)* (pp. 156–163). IEEE.

Betz, J., Betz, T., Fent, F., Geisslinger, M., Heilmeier, A., Hermansdorfer, L., Herrmann, T., Huch, S., Karle, P., Lienkamp, M. et al. (2023). Tum autonomous motorsport: An autonomous racing software for the indy autonomous challenge. *Journal of Field Robotics*, *40*(4), 783–809.

Betz, J., Wischnewski, A., Heilmeier, A., Nobis, F., Stahl, T., Hermansdorfer, L., Lohmann, B., & Lienkamp, M. (2019). What can we learn from autonomous level-5 motorsport? In *9th International munich chassis symposium 2018: chassis. tech plus* (pp. 123–146). Springer.

Caesar, H., Kabzan, J., Tan, K. S., Fong, W. K., Wolff, E., Lang, A., Fletcher, L., Beijbom, O., & Omari, S. (2021). nuPlan: A closed-loop ML-based planning benchmark for autonomous vehicles. arXiv preprint arXiv:2106.11810.

Casas, S., Sadat, A., & Urtasun, R. (2021). MP3: A unified model to map, perceive, predict and plan. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (pp. 14403–14412).

Chen, S., Jiang, B., Gao, H., Liao, B., Xu, Q., Zhang, Q., Huang, C., Liu, W., & Wang, X. (2024). VADv2: End-to-end vectorized autonomous driving via probabilistic planning. arXiv preprint arXiv:2402.13243.

Cheng, J., Chen, Y., & Chen, Q. (2024). PLUTO: Pushing the limit of imitation learning-based planning for autonomous driving. arXiv preprint arXiv:2404.14327.

Dauner, D., Hallgarten, M., Geiger, A., & Chitta, K. (2023). Parting with misconceptions about learning-based vehicle motion planning. In *Conference on robot learning* (pp. 1268–1281). PMLR.

Funke, J., Theodosis, P., Hindiyeh, R., Stanek, G., Kritatakirana, K., Gerdes, C., Langer, D., Hernandez, M., Müller-Bessler, B., & Huhnke, B. (2012). Up to the limits: Autonomous audi TTS. In *2012 IEEE Intelligent vehicles symposium* (pp. 541–547). IEEE.

Heilmeier, A., Wischnewski, A., Hermansdorfer, L., Betz, J., Lienkamp, M., & Lohmann, B. (2020). Minimum curvature trajectory planning and control for an autonomous race car. *Vehicle System Dynamics*. *58*, 1497–1527.

Herrmann, T., Passigato, F., Betz, J., & Lienkamp, M. (2020). Minimum race-time planning-strategy for an autonomous electric racecar. In *2020 IEEE 23rd International conference on intelligent transportation systems (ITSC)* (pp. 1–6). IEEE.

Hu, Y., Li, K., Liang, P., Qian, J., Yang, Z., Zhang, H., Shao, W., Ding, Z., Xu, W., & Liu, Q. (2023a). Imitation with spatial-temporal heatmap: 2nd Place solution for nuPlan challenge. arXiv preprint arXiv:2306.15700.

Hu, Y., Yang, J., Chen, L., Li, K., Sima, C., Zhu, X., Chai, S., Du, S., Lin, T., Wang, W. et al. (2023b). Planning-oriented autonomous driving. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (pp. 17853–17862).

Jiang, B., Chen, S., Xu, Q., Liao, B., Chen, J., Zhou, H., Zhang, Q., Liu, W., Huang, C., & Wang, X. (2023). VAD: Vectorized scene representation for efficient autonomous driving. In *Proceedings of the IEEE/CVF international conference on computer vision* (pp. 8340–8350).

Jung, C., Finazzi, A., Seong, H., Lee, D., Lee, S., Kim, B., Gang, G., Han, S., & Shim, D. H. (2023). An autonomous system for head-to-head race: Design, implementation and analysis; team kaist at the indy autonomous challenge. arXiv preprint arXiv:2303.09463.

Kabzan, J., Valls, M. I., Reijgwart, V. J. F., Hendrikx, H. F. C., Ehmke, C., Prajapat, M., Bühler, A., Gosala, N., Gupta, M., Sivanesan, R. et al. (2020). AMZ Driverless: The full autonomous racing system. *Journal of Field Robotics*, *37*(7), 1267–1294.

Lee, D., Nam, H., Ryu, C., Nah, S., Moon, S., & Shim, D. H. (2024). Enhancing state estimator for autonomous racing: Leveraging multi-modal system and managing computing resources. *IEEE Transactions on Intelligent Vehicles*. *PP(99)*, 1–15.

Li, Z., Li, K., Wang, S., Lan, S., Yu, Z., Ji, Y., Li, Z., Zhu, Z., Kautz, J., Wu, Z. et al. (2024a). Hydra-MDP: End-to-end multimodal planning with multi-target hydra-distillation. arXiv preprint arXiv:2406.06978.

Li, Z., Yu, Z., Lan, S., Li, J., Kautz, J., Lu, T., & Alvarez, J. M. (2024b). Is ego status all you need for open-loop end-to-end autonomous driving? In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (pp. 14864–14873).

Liniger, A., Domahidi, A., & Morari, M. (2015). Optimization-based autonomous racing of 1: 43 scale RC cars. *Optimal Control Applications and Methods*, *36*(5), 628–647.

Na, Y., Kim, S., Seok, J., Ha, J., Kang, J., Lee, J., Jo, J., Lee, J., Kang, H., Lee, J. et al. (2024). AutoKU: An autonomous driving system design for the world's first mass-produced vehicle in multi-vehicle racing environment. In *2024 IEEE Intelligent vehicles symposium (IV)* (pp. 1373–1380). IEEE.

Ni, J., Hu, J., & Xiang, C. (2019). Robust path following control at driving/handling limits of an autonomous electric racecar. *IEEE Transactions on Vehicular Technology*, *68*(6), 5518–5526.

Thrun, S., Montemerlo, M., Dahlkamp, H., Stavens, D., Aron, A., Diebel, J., Fong, P., Gale, J., Halpenny, M., Hoffmann, G. et al. (2006). Stanley: The robot that won the DARPA grand challenge. *Journal of Field Robotics*, *23*(9), 661–692.

Urmson, C., Anhalt, J., Bagnell, D., Baker, C., Bittner, R., Clark, M. N., Dolan, J., Duggins, D., Galatali, T., Geyer, C. et al. (2008). Autonomous driving in urban environments: Boss and the urban challenge. *Journal of Field Robotics*, *25*(8), 425–466.

Weng, X., Ivanovic, B., Wang, Y., Wang, Y., & Pavone, M. (2024). PARA-Drive: Parallelized architecture for real-time autonomous driving. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (pp. 15449–15458).

Werling, M., Ziegler, J., Kammel, S., & Thrun, S. (2010). Optimal trajectory generation for dynamic street scenarios in a frenet frame. In *2010 IEEE International conference on robotics and automation* (pp. 987–993). IEEE.

Wilson, B., Qi, W., Agarwal, T., Lambert, J., Singh, J., Khandelwal, S., Pan, B., Kumar, R., Hartnett, A., Pontes, J. K. et al. (2023). Argoverse 2: Next generation datasets for self-driving perception and forecasting. arXiv preprint arXiv:2301.00493.

Wurman, P. R., Barrett, S., Kawamoto, K., MacGlashan, J., Subramanian, K., Walsh, T. J., Capobianco, R., Devlic, A., Eckert, F., Fuchs, F. et al. (2022). Outracing champion gran turismo drivers with deep reinforcement learning. *Nature*, *602*(7896), 223–228.

Xi, W., Shi, L., & Cao, G. (2023). An imitation learning method with data augmentation and post processing for planning in autonomous driving. https://opendrivelab.com/e2ead/AD23Challenge/Track_4_pegasus_weitao.pdf.

Xin, R., Cheng, J., Liu, H., & Ma, J. (2024). PlanScope: Learning to plan within decision scope does matter. arXiv preprint arXiv:2411.00476.

Zheng, Y., Liang, R., Zheng, K., Zheng, J., Mao, L., Li, J., Gu, W., Ai, R., Li, S. E., Zhan, X. et al. (2025). Diffusion-based planning for autonomous driving with flexible guidance. arXiv preprint arXiv:2501.15564.