I.J. Intelligent Systems and Applications, **2026, 5, 74-108**
Published Online on October 8, 2026 by MECS Press (http://www.mecs-press.org/)
DOI: 10.5815/ijisa.2026.05.05

Hybrid Hierarchical Path Planning and Adaptive Tracking Control for Autonomous Vehicles via HDBHP Optimization and Reinforcement Learning

Lakshmi Narayana*
Department of Computer Science and Systems Engineering, AU-TDR HUB, Andhra University, Visakhapatnam, Andhra Pradesh 530003, India
Department of Artificial Intelligence and Data Science, Seshadri Rao Gudlavalleru Engineering College, Gudlavalleru, Vijayawada, Andhra Pradesh 521356, India
E-mail: ilnarayana1226@gmail.com
ORCID iD: https://orcid.org/0000-0002-5083-6813
*Corresponding author

T. M. N. Vamsi
Department of Computer Science and Engineering, GITAM Deemed to be University, Visakhapatnam, Andhra Pradesh 530045, India
E-mail: mthalata@gitam.edu
ORCID iD: https://orcid.org/0000-0001-6454-3934

Received: March 21, 2026; Revised: May 15, 2026; Accepted: July 17, 2026; Published: October 8, 2026

Abstract: The new hierarchical path planning and tracking control model for autonomous vehicles utilizes hybrid optimization techniques, an adaptive potential field approach, and reinforcement learning to achieve safe, efficient, and adaptive path planning of the vehicles. Initially, a dynamic environment is generated to process path planning and traffic control with obstacles. The global path is computed using the Hybrid Dung Beetle and Hippopotamus Optimization (HDBHP) algorithm for minimum path distance, minimum obstacles, and minimum energy. This path is smoothed using Robust Locally Weighted Regression with Curvature Smoothing (RLWR-CS) to fulfill the vehicle kinematic constraints and obtain smoothness. For real-time local modulation, the Tent Map-Artificial Potential Field (TM-APF) approach is used, which is sensitive to obstacles and uses a chaotic path to plan for a better response. Control is done through Goal-Conditional Q-learning (GCQL) and Prioritized Q-learning (PQL), where decision-making is made with specific goals and prioritized experience replay. An integrated reward function ensures that the path is accurate, safe, fast, and comfortable for the passengers and ensures that the vehicle adapts to the roads from the start to the destination. Simulation results demonstrate that the proposed approach achieves improved Cumulative Rewards of 171.704 at episode 968 and 196.141 at episode 9,251. Also, the presented approach has a lower execution time of 1.65 seconds and outperforms existing approaches such as Double Deep Q-Network (DDQN), Proximal Policy Optimization (PPO), Deep Q-Networks (DQN), and Soft Actor-Critic (SAC).

Index Terms: Autonomous Vehicles, Path Planning, Tracking Control, Hybrid Optimization, Reinforcement Learning.

1. Introduction

Governments, businesses, academic institutions, and the general public are all very interested in Intelligent Vehicles (IVs) because of their potential to revolutionize transportation through advancements in computer technology and artificial intelligence. IV implementation has much potential to improve mobility in crowded metropolitan areas by lowering traffic congestion and minimizing traffic accidents. For autonomous driving to be implemented successfully, it is therefore imperative that planning techniques be safe, reliable, and flexible [1]. One practical solution to the issues of transportation and traffic congestion is autonomous driving technology [2]. However, safety is a significant obstacle to advancing driverless vehicles. Autonomous vehicles' undesirable decision-making practices have the potential to

This work is open access and licensed under the Creative Commons CC BY 4.0 License.

jeopardize life safety and result in significant financial losses. Reinforcement Learning (RL), one of the most sophisticated artificial intelligence methods, has completed several difficult decision-making tasks [3].

To optimize the result, Reinforcement Learning (RL) algorithms learn dynamically through trial and error; they are rewarded for accurate predictions and penalized for inaccurate ones. The most difficult autonomous driving problems, such as planning and decision-making, may be resolved by RL algorithms [4]. The significant potential of RL techniques is demonstrated by their effectiveness in enhancing car-following trajectory control safety, energy consumption, and traffic efficiency. In driving situations with complicated and fluctuating risk levels, cars can operate safely, effectively, and economically with the help of an RL-based adaptive vehicle trajectory control algorithm [5]. Complex system control can be adaptively learned by an adaptive RL algorithm, particularly in situations where there are uncertainties in the environment or system dynamics. Adaptive RL learning-based predictive control is more adaptive to environmental disturbances and has lower computational complexity than traditional model predictive control (MPC) [6].

To ensure driving safety, motion planning is essential. Autonomous Vehicle (AV) motion planning problems have been the subject of numerous studies in recent years. These motion planning algorithms fall into three general categories: learning-based, optimization-based, and graph-based techniques. Most methods are predicated on the premise that traffic information can be accessible and properly predicted, even though they can handle a variety of scenarios [7]. Depending on whether the state space of the configuration space is decoupled, there are two ways to approach motion planning for an autonomous vehicle: direct planning methods and decoupled planning methods [8].

Motion planning and tracking control play an essential role in autonomous vehicle development. The upper-level planned route is the basis for path-tracking control, which applies a specific control algorithm. Because of its breadth, the artificial potential field (APF) method has sparked widespread interest in route or motion planning. Local minima, inaccessible targets, and insufficient safety are only a few of its numerous drawbacks [9]. The global heuristic planning-based artificial potential field (GHP-APF) approach is employed to get the best motion sequence in real time to get around these problems [10]. While following the course, the self-driving cars periodically encounter roadblocks. To avoid dynamic obstacles, AVs must build a viable path in real time using environmental data acquired by the perception module [11]. Automatic driving with limited behaviors is now getting more attention. One of the most prevalent driving behaviors is obstacle avoidance behavior. To achieve completely automated driving as soon as possible, research on autonomous driving under obstacle avoidance behaviors will be helpful [12].

Recent years have seen a rise in the usage of RL for path planning (PP) and dynamic obstacle avoidance (DOA) algorithms [13]. The distribution shift issue resulting in system failure will not affect RL. Numerous studies have employed reinforcement learning (RL) to tackle difficult autonomous driving tasks, such as handling near-accident scenarios, multi-agent interactions, and city driving [14]. Deep reinforcement learning (DRL), one of numerous RL techniques, has been widely employed in the vehicle's specific motion control modules and delivers dependable and positive performance [15]. Despite the autonomous vehicle technology's ongoing development, efficient motion planning and tracking in dynamic and unstructured conditions are still critical. Approaches such as Model Predictive Control (MPC), Deep Q-Networks (DQN), and classical Artificial Potential Fields (APF) are usually unable to achieve real-time consistency because of several shortcomings: Inherent rigidity, poor global path planning, tendency to get stuck in local minima, poor obstacle management, failure to consider kinematic rules, high computational demands, and limitations on adaptability to various conditions. The impetus for this research is to develop a scalable, reliable, and adaptive hierarchical architecture that can safely and efficiently maneuver in dynamic real-world situations with a dynamic road layout and unpredictable traffic. The research aims at unifying hybrid bio-inspired optimization, APF, and deep RL at a multi-level structure.

The research focuses on the creation of a high-performance hierarchical model for autonomous vehicles that encourages safe, effective, and adaptive navigation in dynamic, unpredictable environments. The primary objective is to integrate hybrid optimization, adaptive potential field approaches, and reinforcement learning-based decision making to address challenges in real-time avoidance, smoothness of paths, and quality control decisions. This technique employs a novel Hybrid Dung Beetle and Hippopotamus Optimization (HDBHP) algorithm for global path planning and smoothing with curvature. The goal is to design a strong and elastic structure that improves vehicle autonomy substantially, flexibility in complex traffic environments, and safety and comfort in traveling.

The proposed system uses a hierarchical decomposition principle in contrast to monolithic optimization frameworks. The three control problems addressed as global path planning, local reactive adjustment, and trajectory tracking are considered to be at different temporal orders. The hybrid HDBHP algorithm addresses a global planning problem with multiple objectives in a fixed manner. The TM-APF module is a medium-rate reactive obstacle avoidance algorithm that employs gradient-based navigation with deterministic chaotic perturbations to avoid local minima. The GC-PQL controller operates at the control frequency to produce goal-conditioned tracking actions. The separation guarantees non-overlapping functionality as well as architectural coherence.

The main contribution of the developed model is detailed below,

- *Hybrid HDBHP Optimization Algorithm:* A pioneering approach that combines Dung Beetle and Hippopotamus Optimization is proposed for global path planning to achieve both shortest path length and optimal obstacle avoidance with minimal energy usage.
- *RLWR-CS Smoothing Technique:* Design a Robust Locally Weighted Regression with Curvature Smoothing (RLWR-CS) for achieving smooth, kinematically compliant trajectory planning.

- *TM-APF for Chaotic Local Planning:* A Tent Map-based Artificial Potential Field (TM-APF) allows dynamic obstacle avoidance, overcoming local minima, and improving flexibility.
- *Dual RL Control Mechanism:* combines Goal-Conditional Q-learning (GCQL) and Prioritized Q-learning (PQL) to provide goal-driven, experience-prioritized tracking control and quicker convergence.

The organization of the paper is as follows: Section 2 details related work, Section 3 explains the problem statement and system model, Section 4 explains the proposed methodology, Section 5 details results and discussion, and Section 6 ends with a conclusion.

2. Literature Review

Some of the recent studies related to the proposed research work are reviewed as follows:

Li et al. [16] proposed a decision-making scheme for autonomous vehicles operating in complicated intersections with deep reinforcement learning. By using CRNNs, the authors incorporated the framework to transform visual data from camera sensors to suitable controls for autonomous vehicles (AVs).

The APF-D3QNPER was suggested by Hu et al. [17] in 2023. The Deep Reinforcement Learning (DRL) algorithm applied by Manikandan et al. in 2023 [18] was aimed at enabling autonomous vehicles to reach their destinations in a safe manner, even in dynamic environmental conditions.

In 2023, Tang et al. presented a paper [19] in which they introduced a decision model for self-driving that estimates the entire return distribution through a fully parameterized quantile network (FPQN). In 2024 [20], Zhang et al. proposed an Integrated Planning and Learning (IPL) approach that is based on reinforcement learning to deal with the difficulty of highway speed and lane-changing decisions. In 2022 [21], Yang et al. introduced HMPF, which is a combined system of motion planning. It combines optimization-driven trajectory planning with learning-informed behavior planning to increase motion planning effectiveness.

In 2022 [22] Antonio et al. proposed the more developed solution adv. RAIM, which uses end-to-end multi-agent DRL based on a curriculum through self-play. Li et al. (2022) [23] suggested a deep reinforcement learning algorithm to determine an autonomous driving lane change decision strategy to reduce projected risk. In the work of Albarella et al. (2023) [24], a two-tiered path-planning system for driverless cars was introduced, which combined model-based techniques and deep reinforcement learning. In 2023, Dang et al. [25] suggested an effective event-based Model Predictive Control (eMPC) to deal with the computational problems of model predictive control.

3. Problem Statement and Motivation

Classic methods like DQN, MPC, and classical APF either have heavy computational burdens, are prone to local minima, or are not robust in high-frequency changing traffic situations. Even with extensive advancements in autonomous vehicle (AV) technologies, the motion planning and tracking control system still encounters primary issues in dynamic, unstructured, and crowded environments. Traditional approaches are either not adaptive to fast-changing scenarios, are dependent on pre-trained or offline models with poor generalizability, or require high computational resource consumption not feasible for real-time applications. These difficulties are further witnessed in problematic circumstances, which involve unidentified road users, dynamic barriers, and shifting road circumstances.

Identified Challenges

Global path planning suffers in unpredictable environments, and it tends to be poor or inflexible in terms of trade-offs between the length of the journey, energy consumption, or even safety. In the case of other conventional APF methods, Local Real-Time Obstacle Avoidance exhibits local minima and collision risks. Path Smoothness and Kinematic Feasibility are usually overlooked in optimization-only formulations, resulting in aggressive control actions that sacrifice passenger comfort and tracking stability. Reinforcement Learning-Based Tracking Control tends to be plagued by high training time, poor sample efficiency, and poor transferability between environments. Integrated Decision-Making Frameworks are either computationally too heavy (e.g., MPC + RL hybrids) or do not provide real-time responsiveness under uncertainty.

To overcome these shortcomings, this research explores a new hierarchical path planning and tracking control model for autonomous vehicles that incorporates hybrid optimization, adaptive potential fields, and reinforcement learning algorithms. The novelty of the presented system is in its multi-layered architecture that unifies the HDBHP algorithm for global path calculation, RLWR-CS for path improvement, and TM-APF for real-time obstacle-aware local tuning. Moreover, the use of GCQL and PQL boosts tracking control with adaptive decision-making and efficient experience replay in dynamic traffic situations. The work is novel in its light-weight, adaptive framework combining chaotic dynamics, hybrid intelligence, and real-time reinforcement learning to promote improved path planning and control performance in uncertain, rich-obstacle environments.

4. Methodology

4.1. Overview of the System Model

To drive effectively, an autonomous vehicle must continually design and properly follow the ideal path in real time. Fig. 1 shows a schematic illustration of the motion planning and tracking control issue [26].

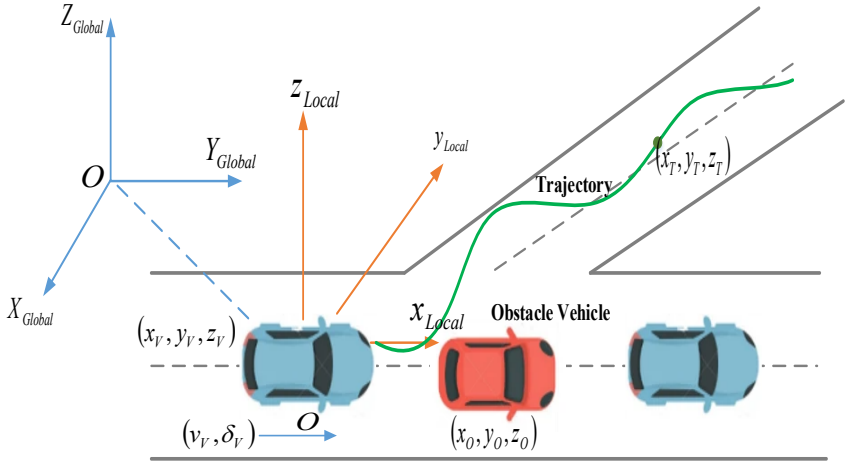


Fig. 1. Schematic of motion planning and tracking control

In the figure, the coordinates denote the inertial coordinate system$X_{Global}, Y_{Global}, Z_{Global}$ and the coordinates indicate the local coordinate system$x_{Local}, y_{Local}, z_{Local}$. Thus, the local and global coordinates represent the motion of an autonomous vehicle. Also, (x_T, y_T, z_T) signifies the position information of waypoints in the path and (x_O, y_O, z_O) represents the position information of the obstacle. On the other hand, (x_V, y_V, z_V) indicates the position information of autonomous vehicles and (v_V, δ_V) signifies driving velocity and steering angle.

Then, consider a kinematic model of an autonomous vehicle [27] in an inertial coordinate system with coordinates (X_F, Y_F) and (X_R, Y_R) represents the center points of the front and rear axles, respectively. Also, the parameter v_R indicates the velocity of the rear axle at the center and δ_F signifies the steering angle of the front axle.

Thus, the velocity of driving an autonomous vehicle is described as Eq. (1).

$$v_R = \dot{X}_R \cos\phi + \dot{Y}_R \sin\phi \tag{1}$$

where $\dot{X}_R$ and $\dot{Y}_R$ represents the position derivatives in directions X and Y in the inertial coordinate system. It is observed from Eq. (1) that the driving velocity is the sum of position derivatives in the driving direction.

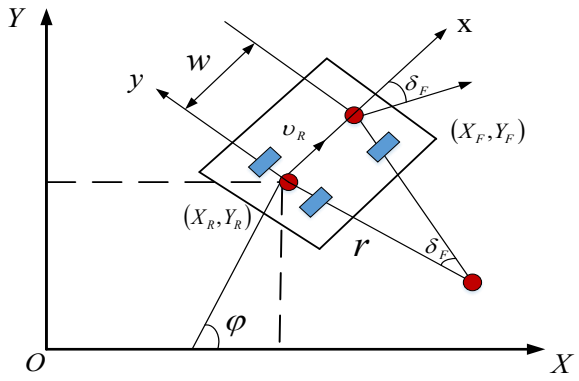


Fig. 2. The kinematic model of an autonomous vehicle

The simplified but commonplace representation of vehicle motion, as shown in Fig. 2, is the kinematic model that is used in autonomous driving, especially during trajectory planning and tracking control at moderate velocities. This model is realistic and represents the behavior of vehicles in the real world by modeling the nonholonomic nature of car-like vehicles in which the motion is limited to the orientation and turning radius of the vehicle. In Fig. 2, W be the wheelbase of an autonomous vehicle and r be the instantaneous turning radius of the rear axle. The kinematic constraints are represented as Eq. (2),

$$\begin{cases} \dot{X}_F \sin(\phi + \delta_F) - \dot{Y}_F \cos(\phi + \delta_F) = 0 \\ \dot{X}_R \sin\phi - \dot{Y}_R \cos\phi = 0 \end{cases} \tag{2}$$

The components of driving velocity in the direction of motion of an autonomous vehicle should be "0". Therefore, the position derivative of the rear axle center is derived as Eq. (3),

$$\begin{cases} \dot{X}_R = \upsilon_R \cos\phi \\ \dot{Y}_R = \upsilon_R \sin\phi \end{cases} \tag{3}$$

Then, based on the geometry of the rear and front axles, the derivatives can be expressed as Eq. (4),

$$\begin{cases} X_F = X_R + w\cos\phi \\ Y_F = Y_R + w\sin\phi \end{cases} \tag{4}$$

The derivatives from the above equations (3) and (4) are imported into equation (2). Then, the instantaneous turning radius and yaw velocity of an autonomous vehicle are expressed as Eq. (5),

$$\begin{cases} \psi = \frac{\upsilon_R}{w} \tan\delta_F \\ \dot{\phi} = \psi \\ r = \frac{\upsilon_R}{\psi} \end{cases} \tag{5}$$

Finally, the kinetic model is constructed by combining the above equations and is described as Eq. (6),

$$\begin{pmatrix} \dot{X}_R \\ \dot{Y}_R \\ \dot{\phi} \end{pmatrix} = \begin{pmatrix} \cos\phi \\ \sin\phi \\ 0 \end{pmatrix} \upsilon_R + \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} \psi = \begin{pmatrix} \cos\phi \\ \sin\phi \\ \tan\delta_F / w \end{pmatrix} \upsilon_R \tag{6}$$

Thus, the obtained kinetic model is applied to the tracking control for an autonomous vehicle. The kinematic model of an autonomous vehicle includes critical behavioral traits required for precise motion representation and control. One critical aspect is the relationship between the wheelbase (w) and steering angle (ϕ), which defines the instantaneous turning radius (r) and yaw rate. Moreover, the velocity at the center of the rear axle (υ_R) is one of the important parameters since it is a simplified yet useful reference for vehicle motion modeling. It is widely utilized in many path following methods like the Pure Pursuit, Stanley Controller, and Model Predictive Control (MPC), due to its simplicity in measurement and the direct relation with the vehicle motion. In addition, local (x, y) and global (X, Y) coordinate frames are conventional in autonomous navigation to ensure precise mapping and positioning within the world. This two-coordinate system is common in most autonomous systems, such as SLAM and global path planning modules.

The suggested methodology works within the specified global and local coordinate systems, with vehicle states $(x_V, y_V, \upsilon_V, \delta_V)$ and obstacle positions (x_O, y_O) as main inputs. The subsequent subsections explain how each part of our hierarchical framework solves the problems identified in the system model. The logical structure of the developed technique is presented in Fig. 3. In contrast to end-to-end reinforcement learning methods, the suggested framework splits the process of navigation into structured global planning, local modulation, and learning-based tracking control layers and thus takes advantage of explicit map information and domain knowledge.

4.2. Optimizing Global Path Planning with a Hybrid Optimization Algorithm

The HDBHP algorithm calculates an optimal sequence of waypoints (x_T, y_T) in the global coordinate scheme. These waypoints are chosen to reduce trip length, avoid barriers (x_O, y_O), and maximize energy efficiency, therefore directly solving the system model's global path planning difficulty.

The biological metaphors are just heuristic inspirations; the algorithm used is a stochastic population-based search algorithm which is intended to strike a balance between exploration and exploitation in a non-convex multi-objective search space.

The proposed HDBHP Optimization algorithm is a combination of the DBO and HOA algorithms. The Dung Beetle Optimization (DBO) algorithm [28] is a new heuristic optimization technique based on the dung beetles' sophisticated movement patterns. Such behaviors include rolling the balls, dancing, foraging, stealing, and reproduction. The Hippopotamus Optimization (HPO) [29] algorithm is inspired by three key behavioral patterns observed in hippopotamuses: Inquisitiveness, roving vigilantism, and escape to the water.

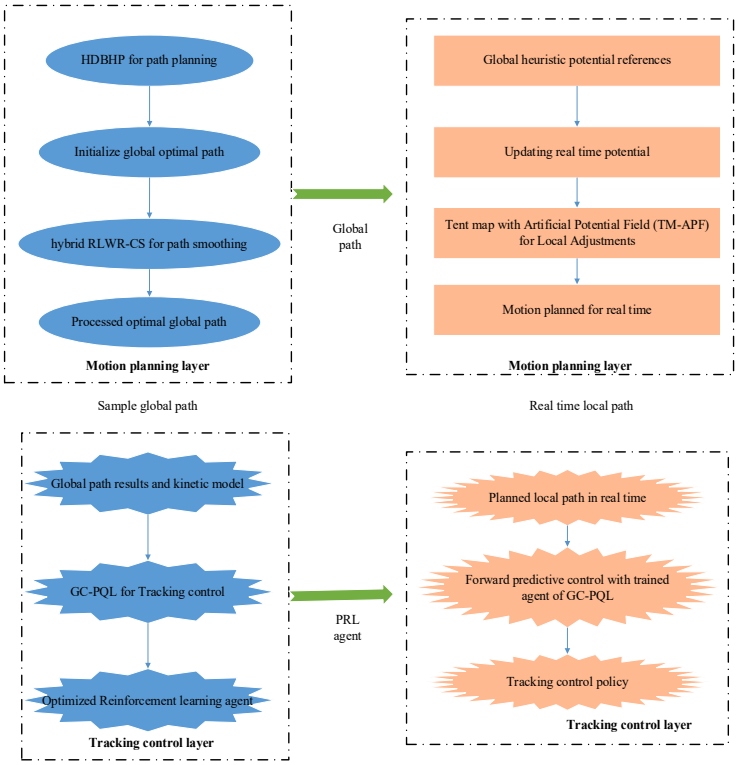


Fig. 3. Logical arrangement of the developed model

The key reason for integrating DBO and HOA is their complementary advantages in exploration and exploitation, which are essential for efficient multi-objective optimization in autonomous vehicle (AV) path planning. DBO, motivated by complex behavioral tactics of dung beetles (e.g., ball rolling, dancing, stealing), is very good at diversifying the search space—it is excellent at exploration, allowing it to find faraway, globally promising areas. Yet, DBO by itself can be devoid of fine-tuning capability in moving toward accurate, high-quality solutions in a bounded environment. HOA (hippopotamus behaviors such as curiosity and evacuation tactics), on the other hand, provides intensive local search (exploitation) dynamics, especially in cases of environmental modification or localized danger (e.g., proximity to obstacles). This is because HOA has a solution refining potential in a given community and it is the perfect complement to the global exploration capabilities of DBO. The HDBHP hybrid achieves balance by combining these two algorithms: DBO's exploratory diversification and HOA's exploitative intensification. Such a two-nature combination is especially appropriate for real-time multi-objective constraints seen in autonomous navigation, where globally optimum path-planning must account for locally variable constraints such as energy, obstacle presence, and real-time environmental changes.

Thus, the proposed HDBHP optimization algorithm is mathematically modeled to optimize global path planning in autonomous vehicles based on multi-objective constraints like distance, obstacles, and energy. Therefore, the suggested multi-objective hybrid optimization allows more comprehensive path planning by enhancing search performance through dynamic exploration and exploitation capabilities. Here, the combined optimization approach integrates the competitive search of DBO with the cooperative search of HOA. The HDBHP algorithm is not merely a sequential combination of two heuristics but a mathematically structured integration of their behavioral models and dynamic search rules. Each behavioral phase from DBO (ball-rolling, reorientation, reproduction, stealing, and foraging) is embedded with HOA's adaptive mechanisms, such as local vigilance and dynamic avoidance. This hybridization allows HDBHP to simulate adaptive behavior that is better aligned with the nature of AV navigation, which involves continuous trade-offs between goal achievement (distance, energy) and constraint satisfaction (obstacle avoidance, safety margins).

The mathematical modeling of the suggested HDBHP optimization algorithm for global path planning is described below in the following steps:

Ball-Rolling Behavior: Dung beetles rely on celestial cues, such as the sun, to maintain a straight path while rolling dung balls. In order to simulate this behavior, the dung beetles in the optimization algorithm are guided to move in a particular direction within the search space. The brightness of light affects the path of the beetle and it determines its direction. The dung beetle is updated on its position as it rolls, depending on this directional movement and the light intensity, which varies. Thus, the ball-rolling behavior is expressed as Eq. (7),

$$u_i(t+1) = u_i(t) + \beta \times m \times u_i(t-1) + d \times \Delta u, \Delta u = |u_i(t) - U^z| \tag{7}$$

Where t denotes the current iteration number, $u_i(t)$ represents the position information of the dung beetle at iterationi^{th} t^{th} , $m \in (0,0.2]$ denotes a constant value indicating the deflection coefficient, d represents a constant value belonging to $(0,1)$, β signifies a natural coefficient that assigns the value -1 or 1, U^z indicates the global worst position and Δu is used to simulate changes in light intensity.

Dancing or Reorientation Behavior: In case this beetle finds that there is a barrier in front of it, it has to dance to look for a new direction. This dance behavior is crucial for dung beetles to allow them to redirect effectively. In the DBO algorithm, this reorientation is simulated by using a tangent function to set a new rolling direction. Once the beetle is straightened, it begins moving the dung ball in the new direction once again. In this phase, the dominant hippopotamus position d_{hip} is updated to find the best route based on the best cost. Thus, the position of the dung beetle is updated using Eq. (8),

$$u_i(t+1) = u_i(t) + \tan(\theta)\left|u_i(t) - u_i(t-1)\right|\left(d_{hip} - i1.u_i\right) \tag{8}$$

Let, d_{hip} denotes the dominant hippopotamus position, $i1$ is denoted as an integer between 1, and θ signifies the deflection angle belonging to $[0, \pi]$.

Reproduction and Spawning Behavior: The dung beetles roll their dung balls to secure locations, hiding them to protect their offspring. It is for this reason that it becomes important for these beetles to choose the right place in which to lay eggs so that their young are protected. From this behavior, the optimization algorithm adopts a boundary selection strategy in modeling the regions where female dung beetles would lay their eggs. Here, the lower and upper bounds of the spawning area are expressed as Eq. (9),

$$\begin{aligned} l_b{}^* &= max(U^* \times (1-K), l_b) \\ u_b{}^* &= min(U^* \times (1-K), u_b) \end{aligned} \tag{9}$$

where U^* denotes the current local best position, $K = 1 - {}^{t}/_{I_{max}}$, I_{max} denotes the maximum iteration number, $l_b$and u_b represent the lower and upper bounds of the optimization problem.

After identifying a suitable spawning area, female dung beetles select brood balls within this region to deposit their eggs. In the DBO algorithm, each female lays only one egg per iteration. It is worth noting that the spawning area boundary is dynamically varied during the optimization process with the value being the major factor K. Consequently, the location of every brood ball is also dynamically changing with every iteration, with the location being described by the formula below, Eq. (10).

$$B_i(t+1) = U^* + d_1 \times (B_i(t) - l_b{}^*) + d_2 \times (B_i(t) - u_b{}^*) \tag{10}$$

where $B_i(t)$ represents the position information of i^{th} brood ball at t^{th} iteration, d_1 and d_2 signifies two independent random vectors of size $1 \times D$ in which D denotes the dimension of the optimization problem.

Foraging Behavior: When some dung beetles grow up and come out of the earth, they look for food. To mimic this natural foraging behavior, the algorithm describes an ideal foraging zone to direct these beetles to the resources. The limits of this foraging area are set as Eq. (11),

$$\begin{aligned} l_b{}^b &= \max\left(U^b \times (1-K), l_b\right), \\ u_b{}^b &= \min\left(U^b \times (1+K), u_b\right) \end{aligned} \tag{11}$$

where U^bdenotes the global best position, $l_b{}^b$ and $u_b{}^b$ represents the lower and upper bounds of the optimal foraging area. The hippopotamus scenarios () are added in this phase to lead additional suitable local searches to attain higher exploitation quality. Then, the position of the small dung beetle is updated as Eq. (12),

$$u_i(t+1) = u_i(t) + R_1 \times \left(u_i(t) + \varsigma - l_b{}^b\right) + R_2 \times \left(u_i(t) + \varsigma - u_b{}^b\right) \tag{12}$$

where $u_i(t)$ denotes the position information of i^{th} small dung beetle at t^{th} iteration, R_1 represents a random number that follows normal distribution and R_2 denotes a random vector belonging to $(0,1)$.

Stealing Behavior: Certain dung beetles, referred to as "thieves," engage in stealing dung balls from other beetles. Here,

U^b represents the optimal food source, suggesting that the area around U^b is the prime spot for competing for resources. The hippopotamus distance D_i and defensive behavior f_{pre} are updated in this phase to improve the stealing behavior. Throughout the iteration process, the position of each thief beetle is updated based on Eq. (13),

$$u_i(t+1) = U^b + C \times h \times \left(\left| u_i(t) - U^* \right| + \left| u_i(t). - U^b \right| \right) \left(\frac{1}{D_i} \right) f_{pre} \tag{13}$$

where $u_i(t)$ denotes the position information of i^{th} thief at t^{th} iteration, h represents a random vector that follows a normal distribution, and C indicates a constant value. The distance () is that of the hippopotamus to the predator. During this time, the hippopotamus adopts a defensive behavior based on the factor f_{pre} to protect itself against the predator. The flowchart of the developed HDBHP model is shown in Fig. 4.

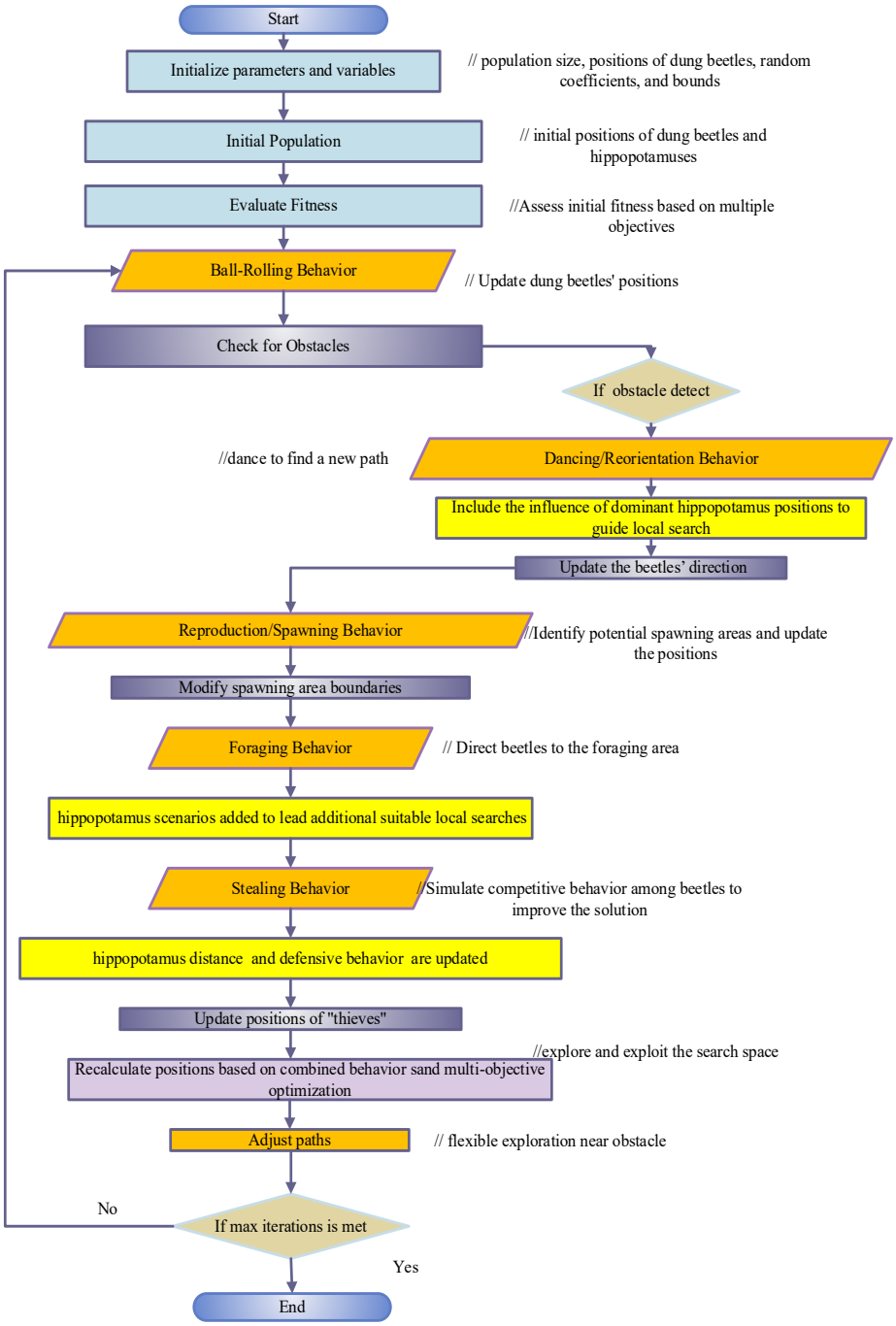


Fig. 4. Workflow representation of the HDBHP model

Multi-Objective Path Optimization: The formulated HDBHP optimization algorithm is aimed at fulfilling the multiple objectives that are needed for the global path planning of autonomous vehicles. This means that the present work adopts a multi-objective optimization approach, which will incorporate distance, obstacles, and energy as the objectives of the path planning to avoid imbalance in the planning process. In real-world path planning for autonomous vehicles, it is not sufficient to optimize a single goal (e.g., the shortest path). The path also needs to consider safety, energy efficiency, and the ability to react in real time to changing conditions. Multi-objective optimization is a compromise between two or more objectives to achieve distance minimization, avoid obstacles, and minimize energy consumption. It minimizes the time and expenses on movements, increases safety and dependability, and takes into account the factors that affect battery or fuel consumption to prolong battery or fuel duration.

The combined behaviors are mathematically modeled to assess multiple goals within each iteration of the process. Position updates, generated by foraging and stealing behaviors, are aimed at allowing the trade-off between exploitation and exploration. The HDBHP algorithm can change paths in real time depending on the weightings of the objectives and thus can be used to make solutions that can respond to real-time constraints and objectives during vehicle navigation. The search rule with variable step is defined by Eq. (14),

$$\zeta_{st} = \begin{Bmatrix} St_{\max} & if\left(D \geq r_{\max}\right) \\ St_{\min} + R\left(\dfrac{D - r_{\min}}{r_{\max} - r_{\min}}\right) \times \left(St_{\max} - St_{\min}\right) & if\left(r_{\max} > D > r_{\min}\right) \\ St_{\min} & if\left(D \leq r_{\min}\right) \end{Bmatrix} \tag{14}$$

where D is denoted as the closest distance between the obstacles and the present position, $r_{\min}$ and $r_{\max}$ represent the minimum and maximum search radius. $St_{\max}$ and $St_{\min}$ indicate the minimum and maximum search steps. The introduced HDBHP is a multi-objective algorithm that can provide efficient overall path planning, which will be critical for AVs to be capable of performing constant adjustments in the real-time environment they inhabit.

4.3. Path Smoothing with Hybrid Robust Locally Weighted Regression and Cubic Splines (RLWR-CS)

In this study, a hybrid RLWR-CS smoothing approach is employed to further smooth the path by reducing the possibility of sharp turns and making the transition from one waypoint to another as smooth as possible. To ensure kinematic feasibility and smoothness, the RLWR-CS technique refines the trajectory between waypoints, considering the vehicle's motion constraints (v_V, δ_V) as specified in the system model schematic. They provide a heuristic reference for the potential field to handle the discontinuities arising from the hybrid optimization. The suggested method allows constant changes of path in the direction of the vehicle's kinematic characteristics.

A. Robust Locally Weighted Regression (RLWR)

RLWR [10] is a versatile regression technique and does not rely on the overall distribution of the data but on local data points to offer a stable smoothing technique. It is particularly useful when path data is not continuous and enables the continuity of the process while maintaining significant characteristics of the path.

Here, the suggested method is applied to generate heuristic potential field references that manage the discrete path points effectively. For each segment of the piecewise path, the waypoints are arranged in a time sequence. Thus, the fitting function is expressed as Eq. (15),

$$F_f(y) = \sum_{j=0}^{g} \gamma_j y^j \tag{15}$$

where γ_j represents the coefficient of j^{th} degree term. Then, the fitting function is minimized using Eq. (16) and is given as,

$$\min_{\gamma} \sum_{i=1}^{M} L(y, y_i) \left(z_i - \sum_{j=0}^{g} \gamma_j y^j\right) \tag{16}$$

where $L(y, y_i)$ represents the distance weight function. Thus, the expression for the distance weight function is described as Eq. (17),

$$L_q(y, y_0) = W\left(\frac{\|y - y_0\|}{q}\right) \tag{17}$$

where $W(t)$ indicates the cubic kernel function and is formulated as Eq. (18),

$$W(t) = \begin{cases} (1 - |t|^3)^3, |t| < 1, \\ 0, otherwise \end{cases} \tag{18}$$

The above Eq. (18) is to be converted into a matrix form and is represented as Eq. (19),

$$\min_{E}(Z - YE)^T L(Z - YE) \tag{19}$$

After converting into matrix form, the coefficients of the polynomial are derived as Eq. (20),

$$E = (Y^T LY)^{-1} Y^T LZ \tag{20}$$

Thus, the proposed RLWR method avoids sharp turning corners and sharp waypoints more effectively than the traditional regression techniques. Then, the residual of the fitting function can be computed using Eq. (21),

$$r_i = z_i - F_f(y_i) \tag{21}$$

Here, the residual weight is calculated by considering the double square function and is expressed as Eq. (22),

$$\xi_i = D(r_i/\chi \cdot p) \tag{22}$$

where p represents the median of the absolute value of residuals, indicatesχ the outlier factor and $D(t)$ signifies the double square kernel function and is modeled as Eq. (23),

$$D(t) = \begin{cases}(1 - t^2)^2, |t| < 1, \\ 0, otherwise\end{cases} \tag{23}$$

The double square kernel function separates waypoints based on the degree of deviation, identifying those significantly beyond the average deviation as sharp waypoints. By setting the residual weight of these sharp waypoints to zero, the creation of sharp turning corners is prevented. Then, the residual weight ξ_i is multiplied by the distance weight function $L(y, y_i)$and the process of regression is repeated. Finally, the global smooth path $GP_{smooth} = \big(y, f(y)\big)|z_{sta} = f(y_{sta}); \big(y, f(y)\big) \in Q^2$that is processed is generated.

B. Cubic Spline Smoothing

Cubic Splines [30] enable production of smooth, continuous and differentiable routes that are highly important in driving safely and efficiently. Cubic splines are polynomials that are defined as piecewise functions passing through a series of waypoints and generate a curve. Path smoothing with cubic splines involves defining a path through waypoints, such as sensor-derived points that the vehicle should pass through or follow. Instead of creating a straight line or simple curve that might cause abrupt turns or changes in direction, cubic splines create a smooth path that minimizes sharp turns. This technique makes sure that the final path is not only free from sharp turns but also has a high degree of smoothness and differentiability, which is important in curvilinear paths.

At first, an additive model is considered and is represented as Eq. (24),

$$v(t_i) = f(t_i) + \varepsilon_i \tag{24}$$

Then, the cubic spline optimization is applied to the additive model and is expressed as Eq. (25),

$$\min_{f \in H} \sum_{i=1}^{k} \big(v(t_i) - f(t_i)\big)^2 + \psi \int (f(t)'')^2 \, dt \tag{25}$$

where ψ represents the smoothing parameter. In formal terms, $_{\mathrm{H}}$ represents a specific Hilbert space of functions; however, in practice, it encompasses all functions for which the smoothness penalty remains finite. It consists of a set of k basis functions and coefficients with $f(t) = \sum_{m=1}^{k} \varphi_m(t) s_m$. Then, the coefficients are determined as in Eq. (26),

$$\min_{s} \|z - \Phi_s\|_2^2 + \psi s^T G s \tag{26}$$

where $\Phi_{s,m} = \Phi_m(t_i)$ and G represents a Gram matrix of the smoothing penalty that is applied to the basis functions and is expressed as Eq. (27),

$$G_{j,m} = \int \varphi_j(t)'' \, \varphi_m(t)'' dt \tag{27}$$

There are multiple standard approaches to selecting these basis functions, each yielding equivalent results but potentially differing in the conditioning of matrix Φ. In every approach, the spline is obtained through ridge regression, with the coefficients calculated using the conventional ridge regression framework. Notably, Φ is of full rank, allowing

it to be reparametrized in terms of the spline function values at the data points.

$$\min_{f} \|z - f\|_2^2 + \psi f^T \Gamma f \tag{28}$$

In the above Eq. (28), when f is a linear function, the penalty $f^T \Gamma f$ will be zero and Γ have a rank $n-2$ with value $\Gamma = \Phi^{-T} G \Phi^{-1}$.

A cubic spline can be interpreted as a prediction from a Gaussian process model. Therefore, the stochastic interpretation of a cubic spline and prediction at arbitrary locations is carried out.

Consider $f(t) = Q(t) + l(t)$ in which $Q(t)$ represents a linear function and $l(t)$ denotes a zero-mean Gaussian process with a covariance function represented as Eq. (29),

$$COV\big(l(t), l(t^*)\big) = h(t, t^*) \tag{29}$$

Then, assume a $n \times 2$ regression matrix Y for fitting a linear trend. Therefore, a usual linear Gaussian process model is expressed as Eq. (30),

$$v = f + \xi = Y\alpha + l + \xi \tag{30}$$

The best linear unbiased estimator for f is given as Eq. (31),

$$\hat{f} = Y\hat{\alpha} + C(C + \sigma^2 K)^{-1}(v - Y\hat{\alpha}) \tag{31}$$

Let, C be the covariance matrix of l, $\hat{\alpha}$ be the generalized least squares estimator of α. The estimator $\hat{f}$ serves as a solution to the minimization problem and is given as Eq. (32),

$$\min_{\alpha, l} \|v - Y\alpha - l\|_2^2 + \psi l^T C^{-1} l \tag{32}$$

In autonomous cars, the resulting spline path provides a reference line that the car is supposed to take into account, taking into account dynamic limitations and possible obstacles. Thus, the suggested RLWR-CS approach is powerful for generating smooth, continuous paths that support stable and efficient navigation in autonomous vehicles. Such methods strike a balance between flexibility, ease of computation, and control stability, making them a widely used technique in path planning for self-driving systems.

4.3. Tent Map with Artificial Potential Field (TM-APF) for Local Adjustments

The Artificial Potential Field (TM-APF) in the Tent Map is a method that involves the real-time sensor data to identify obstacles (x_O, y_O) and adjust the vehicle trajectory dynamically in the local frame within the system model. The TM-APF incorporation provides a novel approach to the autonomous vehicle management system in order to improve its path planning and obstacle avoidance. The potential fields dynamics around obstacles and target goals can be manipulated through the Tent Map, which is a type of map that creates nonlinear paths.

The Tent Map is used as a deterministic perturbation mechanism to prevent the APF gradient descent process from stagnating in local minima. In classical APF, the vehicle motion follows the negative gradient of the potential field, which can converge to undesired equilibrium points. Tent Map provides a means to add controlled chaotic perturbations to the system in order to explore it in a structured fashion. This mechanism plays a role similar to stochastic exploration in optimization algorithms, enabling escape from local minima while preserving directional convergence toward the global objective.

Chaotic Path Generation: The vehicle might follow the control point sequences generated by the Tent Map [31], which offers a variety of non-linear travel alternatives that are sensitive to shifts in the surroundings. The analytical behavior of the Tent Map (a piecewise linear, constant map with a unique maximum) in terms of the power spectrum and invariant density across its chaotic zone is well documented. A series of band-splitting transitions take place in the chaotic zone as the maximum height is decreased, building up to the moment where the chaotic region gives way to the nonchaotic region. The power spectrum and time-correlation values for nonperiodic orbits are computed precisely at and close to the band-splitting sites. The behaviors of the Tent Map according to iteration are the same since it is physically conjugate. The chaotic tent map is given by the equations (33-34).

$$G_{i+1} = F\left(G_i, \vec{\mu}\right) \tag{33}$$

$$F\left(G_i,\vec{\mu}\right)=\begin{Bmatrix} F_l\left(G_i,\vec{\mu}\right)=\vec{\mu}G_i & if\left(G_i<0.5\right) \\ F_r\left(G_i,\vec{\mu}\right)=\vec{\mu}\left(1-G_i\right) & otherwise \end{Bmatrix} \tag{34}$$

Let, $G_i\in[0,1]$ be the normalized interval's control point, for $i\geq 0$. This map only has one control parameter and maps the interval [0, 1] onto itself μ. where $\vec{\mu}\in[0,2]$, the orbit of the complex system is the collection of actual values $G_0,G_1,.....,G_n,.....$. Such an orbit is found for each G_0. The system also displays a diversity of dynamic behaviors, predictable or chaotic depending on the value of the control parameter $\vec{\mu}$. In spite of the fact that the mapping generates chaotic sequences that have certain periodicity in finite accuracy, they have strong statistical properties. The chaotic sequence is periodic as the value $\vec{\mu}$ of the chaotic sequence rises. However, chaotic sequences show high randomness as the value $\vec{\mu}$ gets lower. Then create a series of waypoints with chaotic characteristics by continuing this function and utilizing the outputs as control points. Because of the non-linear path formed by these waypoints, the vehicle may dynamically adjust to changes in its surroundings.

To keep the APF from being locked in local minima (a recognized shortcoming), the Tent Map creates chaotic sequences of control points that disturb the potential field landscape, causing random-like changes in the attraction-repulsion balance. These chaotic disturbances allow the vehicle to explore various micro-paths (i.e., minor deviations) rather than taking a predictable, potentially obstructed course.

Artificial Potential Fields: The car can accelerate around obstacles and modify its course in real time due to the outputs of the Tent Map, which control the attractive and repulsive forces of the APF. The motion sequence is planned using virtual forces in the APF approach. The APF method [32] offers the benefits of real-time planning, great efficiency, and straightforward computation. The APF approach in the suggested TM-APF architecture allows for real-time trajectory modification by continually adjusting the vehicle's travel vector in response to dynamic modifications to the environment, such as the emergence of small or moving impediments on the preplanned global path. When a minor barrier is recognized along the vehicle's route (for example, a person temporarily entering the road or a temporary barricade), the system uses the global heuristic potential field (GHPF), which is precomputed utilizing the global path and stays constant over a short planning horizon. The global path identified by the global algorithm and RLWR-CS process forms the global heuristic potential field (GHPF), as expressed in Eq. (35).

$$u_{gl}\left(a_{pf},b_{pf}\right)=\begin{Bmatrix} 0, & \left(a_{pf},b_{pf}\right)\in p_{sm} \\ \frac{1}{2}\wp p_d^2\left(a_{pf},b_{pf}\right) & \left(a_{pf},b_{pf}\right)\notin p_{sm} \end{Bmatrix} \tag{35}$$

Where p_{sm} is represented as processed global path, and $\left(a_{pf},b_{pf}\right)$ is denoted as any position within GHPF, $\wp$ is considered as heuristic gain constant, and p_d is denoted as global path distance function. $\wp$ establishes the global heuristic potential field's gradient change rate. The unit distance of GHPF changes more as the value $\wp$ of increases. The attracting and repulsing potential fields are then generated using the GHPF. The attractive potential field can be expressed as Eq. (36).

$$u_{atr}\left(a_{pf},b_{pf}\right)=\frac{1}{2}\varepsilon p_{ta}^2\left(a_{pf},b_{pf},a_{ta},a_{ta}\right) \tag{36}$$

Let, ε be the attraction gain constant, and p_{ta} be the distance function of the target point. ε makes the gradient change rate in the attractive potential field. The attractive potential field varies more in the unit distance when the value ε increases. The local repulsive field u_{rpl} (Eq. 37) is dynamically updated with sensor feedback to reflect the current positions of the obstacles.

$$u_{rpl}\left(a_{pf},b_{pf}\right)=\begin{Bmatrix} \frac{1}{2}\gamma\left(\frac{1}{p_{ob}\left(a_{pf},b_{pf}\right)}-\frac{1}{D_{ob}}\right), & p_{ob}\left(a_{pf},b_{pf}\right)\leq D_{ob} \\ 0, & p_{ob}\left(a_{pf},b_{pf}\right)>D_{ob} \end{Bmatrix} \tag{37}$$

Let, γ be the repulsive force constant, p_{ob} be the distance function of obstacle, and D_{ob} be the effect of the range of obstacle. γ defines the rate of change of gradient of the repulsive potential field. The repulsive potential field varies

more in the unit distance when the value of γ increases. The global heuristic potential field, the attractive potential field, and the repulsive potential field are then superposed to form the total artificial potential field. By setting the values of $\wp$, ε and γ reasonably, it is possible to attain the three potential fields' anticipated interaction and combination effect. Moreover, $\wp$ is set to 0.3, ε is set to 0.5 and γ is set to 0.9. The expression of the total artificial potential field is described in Eq. (38).

$$\left\{\begin{array}{l} u_{tl}\left(a_{pf},b_{pf}\right)=u_{gl}\left(a_{pf},b_{pf}\right)+u_{dy}\left(a_{pf},b_{pf}\right) \\ u_{dy}\left(a_{pf},b_{pf}\right)=u_{atr}\left(a_{pf},b_{pf}\right)+u_{rpl}\left(a_{pf},b_{pf}\right) \end{array}\right\} \tag{38}$$

The dynamic potential field u_{dy} is generated dynamically in reaction to variations in the immediate external environment, such as an unforeseen obstruction, while the global heuristic potential field u_{gl} is generated off-line based on the current global map and is fixed. The total field $u_{tl}=u_{gl}+u_{atr}+u_{rpl}$ is recomputed at every time step or sensor update with the help of the Eq. (38). The gradient of the entire potential field yields the autonomous vehicle's motion trajectory in the manner it is described in Eq. (39)

$$\nabla u_{tl}\left(a_{pf},b_{pf}\right) \tag{39}$$

The gradient fall of the entire potential field (Eq. 39) generates a new direction vector for the vehicle, enabling it to subtly change its velocity away from new impediments while remaining focused on the global objective. The autonomous car can travel the planned worldwide route if the outside environment stays the same. To exclude the possible obstacles, the autonomous vehicle is able to generate a local route in real time with the potential field technique, which suggests that the optimal route is blocked by sudden changes in the environment. Lastly, the motion planning layer continually updates the local best path on a rolling basis and outputs it in real time.

The Tent Map is not presented as an ad-hoc innovation but rather as a deterministic perturbation scheme going through the gradient descent dynamics of the artificial potential field. The local planner can be mathematically expressed as Eq. (40).

$$x_{k+1} = x_k - \eta\nabla U(x_k) + \in T(x_k) \tag{40}$$

Where $T(x_k)$ is the chaotic perturbation term. This is a structured perturbation which enables the system to avoid local minima in the potential landscape whilst maintaining convergence properties when the perturbation amplitude is bounded.

Adaptive Path Planning: The APF is used to choose the best trajectory in case of a change in conditions; the Tent Map is used to track and reconsider various different trajectories depending on the information provided by the sensors. The Tent Map uses recalculations of paths to dynamically change the path of the vehicle, using constantly changing sensor data. This allows rapid direction changing by changing the potential fields based on the chaotic properties of the Tent Map. Combined, the TM-APF architecture allows a path-planning algorithm that is less susceptible to obstructions, reacts to environmental changes quickly and avoids predictable and linear paths that may lead to collisions in dynamic environments. In case a small object (like a cone) arises within 1.5 meters of the projected route of the vehicle, the repulsive potential in that place becomes highly exaggerated and the TM-APF re-calculates a new gradient direction by overlaying a modified field that mildly guides the vehicle over the item. As soon as the impediment is no longer sensed, the repulsive potential goes away, and the vehicle goes back to its former path if it is still optimal.

4.4. Goal-conditional with Prioritized Q-learning (GC-PQL) for Tracking Control

Tracking Control based on Goal-conditional with Prioritized Q-learning (GC-PQL) is one of the sophisticated methods proposed to improve the accuracy and flexibility of the tracking control of autonomous vehicles. The Goal-Conditional and Prioritized Q-learning controllers generate control commands (velocity υ_V, steering angle δ_V) to track the planned path, directly implementing the control actions defined in the system model. This method combines goal-conditional learning with prioritized Q-learning to obtain a more efficient control mechanism.

Goal-Conditional Learning (GCL): Goal-conditional learning [33] is a learning algorithm in which the RL agent learns the target conditions of the system in addition to the current state observation. This enables the agent to fine-tune actions with predetermined goals, including the desired flight path, velocity, or position to be achieved. These goals are incorporated into the learning process to make it more flexible and specific so that the vehicle can respond to different cases.

Goal-conditioned reinforcement learning (GCRL), also called goal-oriented RL, is concerned with the problems in which agents are required to reach various goals in an environment or solve a single large-scale task by reaching several goals. In GCRL, the observation (or the state) of the agent is usually accompanied by a goal, which is typically described as a mapped vector from a target state, i.e., $G=\varphi\left(S_{tgt}\right)$. The mapping function $\varphi:\zeta\to G$ is created with a focus on particular duties or merely the identity function. The resulting goal is regarded as being sampled from a distribution q_g. Then, the reward function is defined based on the (state, action, goal) tuple as $R:\zeta\times \mathrm{A}\times G\to\Re$. In such a setting, the agent's policy $\pi:\zeta\times G\to\Omega\left(A\right)$ is trained using Eq. (41) to optimize the expected goal-conditioned return.

$$\max_{\pi} K(\pi)=\mathrm{E}_{q,q_g,\pi}\left[\sum_{t}\delta^{t}R\left(S(t),A(t),G\right)\right] \tag{41}$$

These objectives are to be integrated in a way that makes the agent learn to achieve a range of objectives in an environment. A goal G, which is usually obtained from a target state through a mapping function, is added to the agent's state input. The agent samples goals from a distribution q_g and aims to maximize a goal-conditioned reward , which assesses the degree to which the agent's actions support reaching the objective. The policy $\pi\left(A(t)\middle|S(t),G\right)$ is trained to maximize the expected cumulative return for different goals, enabling the agent to adapt to complex tasks and varying objectives effectively.

Prioritized Q-learning: The first tracking controller is built based on the prioritized Q-learning (PQL) method with the help of the self-learning ability of reinforcement learning algorithms [34]. The Q-learning algorithm and the prioritized experience replay are the main elements of this method. The tracking control layer receives global and local path information from the motion planning layer as training data. The tracking control in an autonomous vehicle involves a sequential decision-making process; current state, g_{t+1} denotes the next state and g_{t+1} p_t signifies the control action.

Then, the optimal control policy is $\pi^*(g_t)$ derived based on Eq. (42),

$$\pi^*(g_t)=\arg\max_{p_t}\left(q(g_t,p_t)+\beta Q^*(g_{t+1})\right) \tag{42}$$

Here, the complete control sequence is generated by solving the optimal tracking control in each step. Then, the target value function is reformulated as Eq. (43),

$$\begin{cases} Q\left(g_t,p_t\right)=q\left(g_t,p_t\right)+\beta\max\limits_{p_{t+1}}Q\left(g_{t+1},p_{t+1};\varepsilon_t\right) \\ Q^*\left(g_t\right)=\max\limits_{p_t}\left(q\left(g_t,p_t\right)+\beta\max\limits_{p_{t+1}}Q\left(g_{t+1},p_{t+1};\varepsilon_t\right)\right) \end{cases} \tag{43}$$

where ε_t represents the set of value factors in the Q Matrix. Then, the updating formula of Q Matrix is expressed as Eq. (44),

$$Q\left(g_t,p_t;\varepsilon_t^{'}\right)\leftarrow Q(g_t,p_t;\varepsilon_t)+\gamma\left[q(g_t,p_t)+\vartheta\max_{p_{t+1}}Q(g_{t+1},p_{t+1};\varepsilon_t)-Q(g_{t+1},p_{t+1};\varepsilon_t)\right] \tag{44}$$

where γ and ϑ indicates the learning rate and discount factor. The learning rate, denoted as γ, determines the extent to which new information overrides old information in the Q-value update. To ensure the convergence of the Q-learning algorithm, γ should satisfy the conditions $\sum_{t=1}^{\infty}\gamma_t=\infty$ and $\sum_{t=1}^{\infty}\gamma_t^2<\infty$. In practice, set the learning rate to a small constant value (e.g., γ = 0.1), which balances convergence speed and stability. Alternatively, a decaying learning rate or an adaptive scheme based on state-action visit counts can be employed to further guarantee convergence. Then, the reward function of the tracking control is set by considering the requirements of safety, accuracy, comfort, and rapidity. Therefore, the reward function is expressed as Eq. (45),

$$q\left(g_t,p_t\right)=c_{Acc}\cdot f_{err}\left(g_t,p_t\right)+c_{Saf}\cdot f_{Saf}\left(g_t,p_t,K_{dc}\right)+c_{rap}\cdot f_{velo}\left(g_t,p_t,\upsilon_{\max}\right)+c_{cc}\cdot f_{dc}\left(p_t\right) \tag{45}$$

where c_{Acc} represents the accuracy coefficient, c_{Saf} denotes the safety coefficient, c_{rap} indicates the rapidity coefficient, c_{cc} denotes the comfort coefficient, f_{err} is the tracking error function, f_{Saf} indicates the safety guaranteed function, f_{velo} is the driving velocity function and f_{dc} signifies the driving comfort evaluation function, respectively.

The reward components are bounded to maintain numerical stability and interpretability. The theoretical maximum episodic reward is achieved when tracking error approaches zero, safety margins remain above threshold, velocity remains within optimal range, and jerk is minimized. Conversely, large penalties are assigned for collisions, unsafe distances, excessive jerk, or tracking deviations. This bounded design ensures that the cumulative reward provides a meaningful performance indicator across different algorithms.

In this section, two distinct discount-related parameters are introduced. The first, denoted as β, is the standard reinforcement learning discount factor used in computing the Q-value updates to balance current and future rewards. The second, often represented as ϑ is used in the prioritized experience replay mechanism to determine the influence of sample priority on replay probability. While β ensures temporal reward consistency, ϑ tunes the learning stability and convergence during the experience replay process.

Since safety, rapidity, accuracy, and comfort are mutually conflicting objectives, a hierarchical weighted strategy is adopted in the reward formulation. In a practical autonomous driving case, accelerated speed can decrease safety margins and jerk, thus influencing the comfort of the passengers. Thus, the coefficients of rewards are not allocated equally but on a priority basis.

Safety is granted the first priority to ensure no collision and minimum obstacle clearance. The weight of accuracy is rated second, so as to achieve accuracy in tracking the path. Comfort, which is gauged by jerk and steering smoothness, has moderate weight to prevent aggressive behavior. Rapidity (vehicle speed) is attributed with a relatively low weight so as to avoid unsafe acceleration.

The weighting coefficients meet the following constraint given in Eq. (46):

$$\omega_{safety} > \omega_{accuracy} > \omega_{comfort} > \omega_{speed} \tag{46}$$

This makes the improvement of speed only rewarded if the safety and tracking accuracy are not compromised. Also, penalty terms are enabled in case of jerk, steering rate or minimum safety distance going above set limits. This is a conditional mechanism of punishment, to ensure that the agent does not learn dangerous high-speed behaviors. Therefore, the reward balancing system ensures that the fast motion is reached in the conditions of safe, stable, and comfortable driving.

The use of experience replay to train the Q matrix depends on the fact that training samples are stored in a replay buffer and sampled in accordance with particular requirements, which is aimed at maximizing the matrix. Prioritized experience replay (PER) is used to facilitate the Q-learning algorithm to improve training efficiency. In PER, priority is based upon the temporal difference (TD) error which is the difference between the target value and the actual value functions of the individual experience samples. The higher the TD error, the greater the priority assigned to that experience sample. The priority of the experience replay is positively correlated with the TD error and is formulated as Eq. (47),

$$\begin{cases} TD(g_t, p_t) = \left| Q^{Tar}(g_t, p_t) - Q^{Act}(g_t, p_t) \right| \\ Q^{Tar}(g_t, p_t) = q(g_t, p_t) + \vartheta \max\limits_{p_{t+1}} Q(g_{t+1}, p_{t+1}; \varepsilon_t) \\ Q^{Act}(g_t, p_t) = Q(g_t, p_t; \varepsilon_t) \end{cases} \tag{47}$$

Then, the probability of each sampled experience replay can be estimated based on Eq. (48),

$$P_t = \frac{\left(TD(g_t, p_t) + \sigma\right)^{\chi}}{\sum_{c=1}^{c=m_{buffer}} \left(TD(g_c, p_c) + \sigma\right)^{\chi}} \tag{48}$$

where χ denotes the priority factor and the parameter is used to avoid division by zero.σ In this study, the quality of Q matrix training is assessed using the generalized correlation coefficient, which measures the similarity of matrices over various time points. Training is considered complete when the coefficient approaches a value of 1. Equation (51) defines a generalized correlation coefficient $\Omega\left(Q_{cur}, Q_{las}\right)$ which quantitatively measures the similarity between the current Q-matrix (Q_{cur}) and the last Q-matrix (Q_{las} *s*) during the training process. The structure is given as Eq. (49).

$$\Omega\left(Q_{cur}, Q_{las}\right) = \frac{tq[\mathrm{cov}(Q_{cur}, Q_{las})]}{\{tq[\mathrm{cov}(Q_{cur})] \times tq[\mathrm{cov}(Q_{las})]\}^{1/2}} \tag{49}$$

where $tq(A)$ indicates the mathematical trace function of the matrix A, Q_{cur} and Q_{las} represents the current and last Q matrix. The form of Equation (50) is motivated by the Pearson correlation coefficient, adapted to matrix format. It captures the global similarity between the latest and past Q-matrices by comparing their covariance, standardized by their variances (traces of covariance matrices). This gives a strong, scalar measure of convergence in Q-learning, guaranteeing that the

policy learned has settled globally. Use of the trace function guarantees sensitivity to change in all state-action pairs, which makes it both a good and understandable measure of progress in training. Thus, the PQL helps optimize path following by assigning priority to more important experiences during training. By using prioritized experience replay (PER), PQL improves learning efficiency by focusing on experiences with higher errors, which are more valuable for refining the Q function. This results in better tracking control that is more accurate and adaptive and enables the vehicle to track the planned paths effectively, taking into account such factors as safety, speed, and comfort.

Table 1. Notation and its description

Notations	Description	Notations	Description
$X_{Global}, Y_{Global}, Z_{Global}$	Inertial coordinate system	ε	attraction gain constant
$x_{Local}, y_{Local}, z_{Local}$	Local coordinate system	p_{ta}	target point's distance function
(x_T, y_T, z_T)	Position information of the waypoints in the path	γ	repulsion gain constant
(x_O, y_O, z_O)	Position information of the obstacle	p_{ob}	Obstacle's distance function
(x_V, y_V, z_V)	Position information of autonomous vehicles	D_{ob}	impact of the obstacle's range
(υ_V, δ_V)	Driving velocity and steering angle	u_{dy}	dynamic potential field
(X_F, Y_F) and (X_R, Y_R)	Center points of the front and rear axles	u_{gl}	global heuristic potential field
υ_R	Velocity of the rear axle at the center	G	goal
δ_F	Steering angle of the front axle	$R(S(t), A(t), G)$	goal-conditioned reward
$\dot{X}_R$ and $\dot{Y}_R$	Position derivatives in directions X and Y in the inertial coordinate system.	$\pi : \zeta \times G \to \Omega(A)$	agent's policy
w	The wheelbase of an autonomous vehicle	q_g	distribution
r	Instantaneous turning radius of the rear axle	E	expectation function
t	current iteration number	$q(g, p)$	reward function
$u_i(t)$	position information	β	discount factor
d	constant value	π	follow-up optimal policy
β	natural coefficient that assigns the value -1 or 1	g_t	current state
U^z	global worst position	g_{t+1}	next state
Δu	simulate changes in light intensity.	p_t	control action
d_{hip}	dominant hippopotamus position	$\pi^*(g_t)$	optimal control policy
i1	integer between 1	ε_t	A set of value factors in the Q Matrix
θ	deflection angle	γ and ϑ	learning rate and discount factor
U^*	current local best position,	$\ddot{\mu}$	control parameter's value
l_b and u_b	lower and upper bounds	p_{sm}	processed global path,
$B_i(t)$	position information	Y	regression matrix
d_1 and d_2	Two independent random vectors	C	covariance matrix
D	dimension	$\hat{\alpha}$	generalized least squares estimator
R_1, and R_2	random vector	$\hat{f}$	estimator
U^b	optimal food source	$G_i \in [0,1]$	normalized interval's control point
D_i	hippopotamus distance	$D(t)$	Double square kernel function
f_{pre}	defensive behavior	ξ_i	residual weight
C	constant value	ψ	smoothing parameter
D	the closest distance between the obstacles and the present position	H	a specific Hilbert space of functions
$r_{\min}$ and $r_{\max}$	minimum and maximum search radius	k	basis functions
$St_{\max}$ and $St_{\min}$	minimum and maximum search steps	G	Gram matrix
γ_j	coefficient of j^{th} degree term	f	a linear function
$F_f(y)$	fitting function	$Q(t)$	linear function
$L(y, y_i)$	distance weight function	$l(t)$	zero-mean Gaussian process with a covariance function
$W(t)$	cubic kernel function	$\wp$	heuristic gain constant,
p	median of the absolute value of residuals	p_d	global path distance function
χ	outlier factor	f_{Saf}	Safety guaranteed function
c_{Acc}	accuracy coefficient,	f_{velo}	Driving velocity function
c_{Saf}	safety coefficient	f_{dc}	Driving comfort evaluation function

c_{rap}	rapidity coefficient	χ	priority factor
c_{cc}	comfort coefficient	$tq(A)$	mathematical trace function of the matrix
f_{err}	tracking error function		

The proposed GC-PQL model is an improvement in tracking control of AVs, which brings together goal-conditional learning and prioritized Q-learning. The goal-conditional learning allows the RL agent to incorporate conditions associated with goals, like path or velocity, into the model. This makes the agent capable of changing its operations to achieve different predefined objectives, thus making it versatile about the environment and tasks. In parallel, prioritized Q-learning (PQL) improves the training performance by prioritizing the experience replay by the experiences with the higher TD errors. This makes it possible for the agent to concentrate on more informative training patterns, thus making learning more efficient. The tracking control mechanism is based on a Q-matrix that compares the expected accumulated reward of actions with a reward function that considers path accuracy, safety, speed, and comfort. By updating and enhancing the Q-matrix over time, the model learns how to optimize its policy to achieve high accuracy in the vehicle's path planning while at the same time coping with the traffic dynamics. By considering these factors, GC-PQL offers a stable method of control tracking, and it is possible to tune the controller parameters to allow AVs to move safely and comfortably while demonstrating the best path-following performance. Despite the continuous reinforcement learning algorithms Soft Actor-Critic and Twin Delayed Deep Deterministic Policy Gradient being designed to use the continuous action space, the hierarchical architecture presented has split continuous geometric path generation (managed in TM-APF and RLWR-CS) and discrete supervisory tracking control. The GC-PQL module is a high-level policy chooser among limited control commands instead of a low-level torque controller. The reason why discrete Q-learning was chosen is that it has a lower computational complexity, converges quicker in a small dimensional state space, and can be deployed more dependably in embedded systems. The notation and its description are detailed in Table 1.

Action Space Discretization for Tracking Control

The tracking control problem inherently involves continuous control variables, namely steering angle δ and longitudinal velocity v. As classical Q-learning is applied to discrete action spaces, the continuous control actions are transformed into finite action spaces and the GC-PQL is applied.

The steering angle $\delta \in [-\delta max_{max}]$ is uniformly quantized into discrete levels N_δ defined as in Eq. (50):

$$\delta_i = -\delta\delta_{max} \tag{50}$$

Where the quantization step size is defined in Eq. (51).

$$\Delta_\delta = \frac{2\delta_{max}}{N_\delta - 1} \tag{51}$$

Similarly, the velocity $v \in [vmax_{min}]$ is discretized into N_v levels. The final action space is constructed as a Cartesian product as expressed in Eq. (52).

$$A = \{(\delta_i, v_j)\} \tag{52}$$

In this study, $N_\delta = 1$ and $N_v = 5$ resulting in 55 discrete control actions. This discretization provides sufficient resolution for accurate path tracking while maintaining computational efficiency and convergence stability.

4.5. Real-Time Integration Mechanism of the Hierarchical Framework

The proposed framework does not use APF as a safety shield reward for RL, nor does RL adapt the parameters of the potential field. Instead, TM-APF operates as a deterministic local trajectory modulation layer, and GC-PQL acts purely as a tracking controller. This strict functional separation ensures that geometric obstacle avoidance and learning-based control remain mathematically and computationally decoupled.

This section explains the real-time hierarchical integration strategy adopted in the proposed framework. The architecture follows a closed-loop multi-layer structure operating at different temporal scales.

At the upper level, the Hybrid Dung Beetle and Hippopotamus Optimization (HDBHP) algorithm computes a globally optimal waypoint sequence by minimizing path length, obstacle proximity, and energy consumption. RLWR-CS smoothing is used to refine the obtained path to make it kinemically feasible and with continuity of curvature. This is a smooth trajectory which is used as a long horizon reference path and is only updated when there is a significant change in the environment.

At the intermediate level, the Tent Map-Artificial Potential Field (TM-APF) module performs real-time local path modulation. Using continuous sensor feedback, dynamic obstacles are detected and incorporated into the repulsive potential field. The global heuristic potential field derived from the optimized global path acts as an attraction backbone, while the Tent Map introduces controlled chaotic perturbations to avoid local minima. The total artificial potential field is recomputed at every planning cycle, and its gradient generates a locally adjusted reference trajectory. If no sudden

obstacle is detected, the vehicle follows the global reference path; otherwise, the local path is modified temporarily to ensure safe avoidance and then smoothly returns to the global trajectory.

At the lower level, the Goal-Conditional Prioritized Q-learning (GC-PQL) controller executes tracking control in real time. The localized trajectory and the present state and objective of the vehicle are sent to the controller. Based on the Q-value estimation, optimal control actions (steering angle and velocity) are generated at each time step. The reward function incorporates tracking accuracy, safety margin, speed regulation, and passenger comfort. Through prioritized experience replay, the controller continuously improves policy convergence while maintaining stability.

The interaction among layers follows a rolling-horizon mechanism. The global planner provides a strategic reference path, the TM-APF module refines it in response to short-term environmental variations, and the GC-PQL controller converts the updated trajectory into executable control commands. Feedback from vehicle states and environment perception continuously propagates upward, enabling adaptive replanning when required. This hierarchical breakdown guarantees computational effectiveness, real time responsiveness, and stability in dynamic driving settings.

Architectural Necessity and Hyperparameter Decoupling Strategy

The proposed architecture is not a stacked aggregation of heterogeneous methods but a strict hierarchical decomposition of three mathematically distinct problems operating at different temporal scales. The HDBHP module solves a non-convex multi-objective global optimization problem in waypoint space and operates at low frequency. RLWR-CS is not an independent planner but a deterministic geometric constraint-enforcement step ensuring curvature continuity and kinematic feasibility. The TM-APF module performs high-frequency local reactive obstacle modulation using gradient descent with bounded chaotic perturbation to avoid local minima. The GC-PQL controller addresses sequential decision-making for trajectory tracking under safety, comfort, and velocity constraints. Each layer produces a reference signal for the lower layer without circular feedback, thereby preventing error accumulation.

Hyperparameter tuning is conducted independently within each layer due to unidirectional data flow (Global → Local → Control). Global planner parameters are tuned offline using sensitivity analysis on path length, energy, and obstacle clearance metrics. RLWR-CS parameters are selected via curvature variance minimization. TM-APF gains and chaotic coefficients are tuned based on safety margin and oscillation stability criteria. The convergence analysis of Q-matrix and reward stability is used to optimize GC-PQL parameters. This decoupled tuning plan avoids the cross-layer instability and provides computational tractability.

4.6. Architectural Coherence and Design Rationale

The integration between HDBHP, RLWR-CS, TM-APF, and GC-PQL is based on problem decomposition rather than methodological stacking. Autonomous navigation involves three fundamentally different mathematical formulations:

- global multi-objective optimization in non-convex space,
- local gradient-based reactive obstacle avoidance under real-time constraints, and
- sequential control decision-making under uncertainty.

A standard Global Planner + Local MPC/RL architecture typically merges (2) and (3), which increases computational load and may cause instability under dynamic obstacle conditions. In contrast, the proposed framework isolates geometric feasibility (RLWR-CS), reactive modulation (TM-APF), and learning-based tracking (GC-PQL), ensuring functional non-overlap. The ablation study (Table 12) empirically confirms architectural necessity, as removal of any module results in measurable degradation in adaptability, safety margin, smoothness, or convergence stability. Therefore, the framework improves robustness and interpretability rather than introducing unnecessary complexity.

5. Results and Discussion

In this section, the performance of the presented motion planning and tracking control framework is validated through numerical simulation. Here, the global heuristic-based Tent Map-Artificial Potential Field (TM-APF) method and Goal-Conditional Q-learning (GCQL) based forward predictive control technique are compared with some existing approaches.

5.1. Performance Evaluation Metrics

To ensure rigorous and reproducible evaluation, the following quantitative metrics are used:

A. Cumulative Reward

Cumulative reward measures provide reinforcement learning performance by adding step-by-step rewards in an episode, as defined in Eq. (53).

$$R_{total} = \sum_{t=0}^{T} r_t \tag{53}$$

B. Success Rate (%)

The success rate is defined as the percentage of goal-reaching trials without collision and the collision rate represents the ratio of collision occurrences to the number of episodes, as expressed in Eq. (54).

$$Success\ Rate = \frac{Successful\ Trials}{Total\ Trials} \times 100 \tag{54}$$

C. Collision Rate

To measure the smoothness of the path, the trajectory jerk, which is the third time-derivative of position, is used to assess passenger comfort and stability of control, as defined in Eq. (55).

$$Collision\ Rate = \frac{Number\ of\ Collisions}{Total\ Episodes} \tag{55}$$

Lower values indicate better safety performance.

D. Path Smoothness (Jerk)

Trajectory smoothness is quantified using jerk, as defined in Eq. (56):

$$Jerk = \frac{d^3x}{dt^3} \tag{56}$$

The reported value corresponds to the average absolute jerk over the trajectory.

E. Safety Margin

Safety margin is defined as the minimum Euclidean distance between the vehicle and surrounding obstacles during an episode.

F. Normalized Performance Score (%)

The Normalized Performance Score (%) is defined as a normalized composite performance index, as expressed in Eq. (57):

$$CR = \omega_1 A + \omega_2 S + \omega_3 C + \omega_4 E \tag{57}$$

where represents tracking accuracy, S represents safety index, represents C comfort index, E denotes efficiency.

5.2. RL Performance Analysis

Fig. 5 shows that the gradient of convergence of the reinforcement learning behavior of the proposed GC-PQL controller is achieved among training sessions. The cumulative reward (blue) grows slowly with the progression of training, which suggests better optimization of the policy and learning of the dynamics of the environment. At the same time, the success rate (orange curve) is steadily growing upwards, and it is close to saturation at an early episode, which proves the ability to complete the task better.

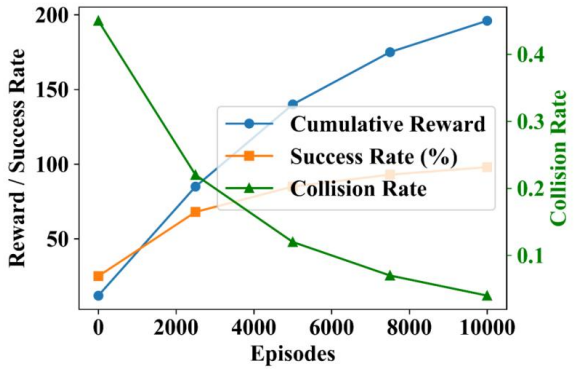


Fig. 5. Learning performance evaluation of the proposed GC-PQL framework

On the other hand, the collision rate (green curve) which is plotted on the secondary vertical axis declines greatly with training. This shows a high level of safety performance and low rates of collision because of the refinement of the policy. The negative correlation between cumulative reward and collision rate is the evidence that the reward structure is used to suitably punish the unsafe actions and reward successful navigation. In general, these results confirm the stable convergence, increased safety, and efficient execution of the tasks of the suggested hierarchical structure.

5.3. Reward Normalization and Interpretation

To be able to compare the results and interpret them, the cumulative reward values are normalized by the maximum possible value of episode returns. The normalized score at the performance level is calculated as:

$$R_{norm} = \frac{R - R_{min}}{Rmin_{max}} \tag{58}$$

where R is the average episodic return, and R_{max}, R_{min} represent the theoretical upper and lower reward bounds defined by the reward function in Eq. (58).

According to this normalization, the performance score of the proposed method is 97.4%, which is much higher than that of DDQN (82.1%), PPO (85.3%), SAC (88.7%), and DQN (78.5%). This ascertains that it is not merely the extent of raw rewards that is enhanced, but also the relative optimality of the policy.

5.4. Computational Environment and Simulation Setup

Table 2 shows the Computational and Simulation Setup Used for Execution Time Evaluation. The reported execution time of 1.65 seconds corresponds to the total computation time required to complete one full navigation episode under the proposed hierarchical framework, including global optimization (HDBHP), RLWR-CS smoothing, TM-APF local adjustment, and GC-PQL tracking control. It is important to clarify that global path optimization is not executed at every control cycle; rather, it operates at a lower frequency, while the local TM-APF adjustment and GC-PQL tracking controller update at significantly higher rates within the episode. In practical autonomous driving systems, hierarchical planning architectures typically separate long-horizon global planning from high-frequency control loops. Therefore, although the full episode execution time is 1.65 seconds on a standard CPU without GPU acceleration, the real-time control decisions occur at a much finer temporal resolution, satisfying practical real-time requirements. Environmental variability is incorporated in the simulation through randomized obstacle motion patterns, state perturbations, and stochastic environmental disturbances. Although explicit weather and lighting models are not separately simulated, the framework is evaluated under randomized dynamic conditions that introduce uncertainty in perception and motion, thereby testing robustness to environmental variability.

Table 2. Computational and simulation setup used for execution time evaluation

Category	Parameter	Specification
Computational Platform	Device Type	Personal Computer
	Processor	Intel Core i5-5200U CPU @ 2.20 GHz
	System Architecture	64-bit operating system, x64-based processor
	Installed RAM	4 GB
	Programming Environment	Python
Simulation Configuration	Environment Type	Dynamic obstacle navigation scenario
	Map Representation	2D continuous navigation space
	Obstacle Model	Static and dynamically appearing obstacles
	Maximum Iterations	1000
	Population Size (Optimization)	20
	State Dimension	6
	Forward Prediction Horizon	8 steps
	Exploration Factor	0.8
	Discount Factor	0.9
Execution Context	Reported Execution Time	1.65 seconds
	Measurement Definition	Total computation time per full navigation episode
	Planning Structure	Hierarchical global + local planning
	Local Adjustment	Real-time potential field update

Table 3 shows the Simulation Platform and Environment Setup and Table 4 shows the Obstacle and Agent Configuration Details.

Table 3. Simulation platform and environment setup

Category	Parameter	Specification
Simulation Platform	Implementation Language	Python
	Environment Type	Custom dynamic navigation simulator
	Simulation Dimension	Two-dimensional continuous space
Road Scenario	Navigation Context	Urban-style dynamic environment
	Traffic Type	Mixed static and dynamic obstacles
	Motion Model	Kinematic vehicle model
Map Configuration	Map Size	100 × 100 m bounded space
	Goal Region	Fixed target with dynamic path adaptation
	Path Representation	Continuous trajectory planning
Obstacle Configuration	Static Obstacles	Random spatial distribution
	Dynamic Obstacles	Time-varying positions and velocities
	Obstacle Density	Medium-complexity environment
	Minimum Safety Distance	1.2 m
Agent Interaction	Autonomous Vehicle	Single ego agent
	Environment Update	Real-time obstacle perturbation
	Collision Model	Distance-based constraint

Table 4. Obstacle and agent configuration details

Category	Parameter	Value
Autonomous Agents	Ego Vehicle	1
	Control Policy	Reinforcement learning-based tracking
Static Obstacles	Number of Obstacles	15
	Distribution	Random spatial placement
	Motion	Stationary
Dynamic Obstacles	Number of Obstacles	5
	Motion Pattern	Time-varying position updates
	Interaction Model	Collision avoidance constraint
Environment Complexity	Total Obstacles	20
	Scenario Type	Medium-density dynamic environment
	Safety Margin	1.2 m

5.5. Simulation Results of Motion Planning

The path planning visualization of autonomous driving presents the surrounding of the vehicle, the roads, obstacles as well as the dynamic objects such as pedestrians or other vehicles. The visualization emphasizes the intended route, which is commonly depicted as a sequence of linked checkpoints, or a curved line, so that the vehicle will not collide with other vehicles yet will not break the traffic regulations. Thus, by visualizing several constraints like speed limits, lane changes, and sudden obstacles, the engineers and operators can evaluate decision-making, fine-tune algorithms, and ensure the system reacts predictably to complex scenarios. The details of parameter settings and their values for GCQL are shown in Table 5.

Table 5. Parameter setting of GCQL method

Parameter	Value
Population size	20
Learning rate	0.05
Discount factor	0.9
Dimension	6
Exploration factor	0.8
Exploration attenuation rate	0.95
Forward Prediction Step	8
Prediction attenuation factor	0.25
Number of obstacles	Randomly generated
Maximum iterations	1000

The graph representing visualization during path planning in autonomous vehicles is illustrated in Fig. 6.

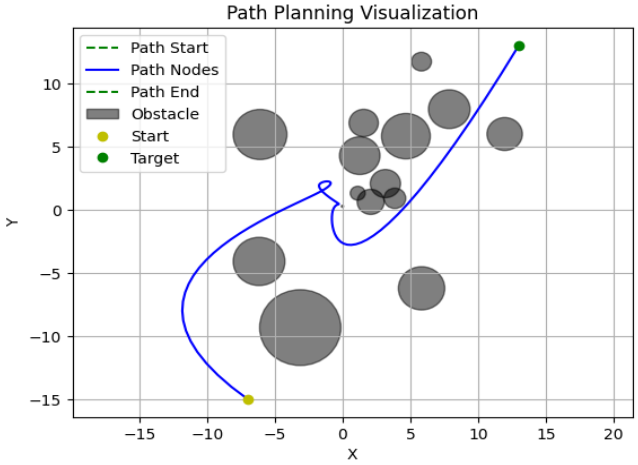


Fig. 6. Global Paths planned visualization without sudden obstacles

The graph illustrates a path-planning scenario for an autonomous vehicle navigating a cluttered environment. The black circles depict the obstacles and the blue line the desired path of the planned trajectory between a starting point (yellow marker) and a goal point (green marker). The route does not collide and is as smooth and efficient as possible, which proves the application of motion-planning algorithms. The car is capable of dynamically computing safe paths by considering the position and size of objects in order to have an obstacle-free path. This diagram illustrates why motion planning is so important in facilitating autonomous systems to perform their functions safely within a complicated and obstructed environment.

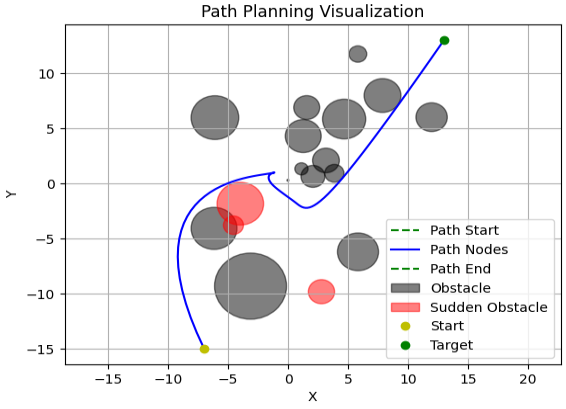


Fig. 7. Sudden obstacles visualization global path planning

In the case of an unforeseen obstacle in the course of vehicle movement, the autonomous system should run a real-time local path replanning in order to prevent a collision. This uncertainty is bound to cause an increase in the total driving distance. Nevertheless, as the issue of safety is of the highest priority, a more lengthy route is a decent compromise. The difficulty is to develop a motion plan that enables the vehicle to move in tight and restricted areas and provide safe and precise tracking control. The abrupt appearance of the impediments does not only increase the threat of collisions but also introduces complexity to the planning and control procedure, becoming an essential element of the autonomous vehicle work.

Fig. 7 illustrates the path planning of an autonomous vehicle that has just met an unexpected obstacle in the world. The black circles symbolize fixed obstacles, whereas the red circle symbolizes an obstacle that is going to appear dynamically. The blue line represents the revised route that is computed in real-time to prevent unforeseen challenges during the movement between the starting point and the destination point. The introduction of the red obstacle is a sudden event that brings to focus the necessity of the vehicle to conduct real-time local path replanning to be safe. This usually takes the form of increasing the driving range with the aim of emphasizing on collision avoidance. The graph highlights the issues of dynamic environments in motion planning and reactive algorithms to navigate safely in uncertain conditions.

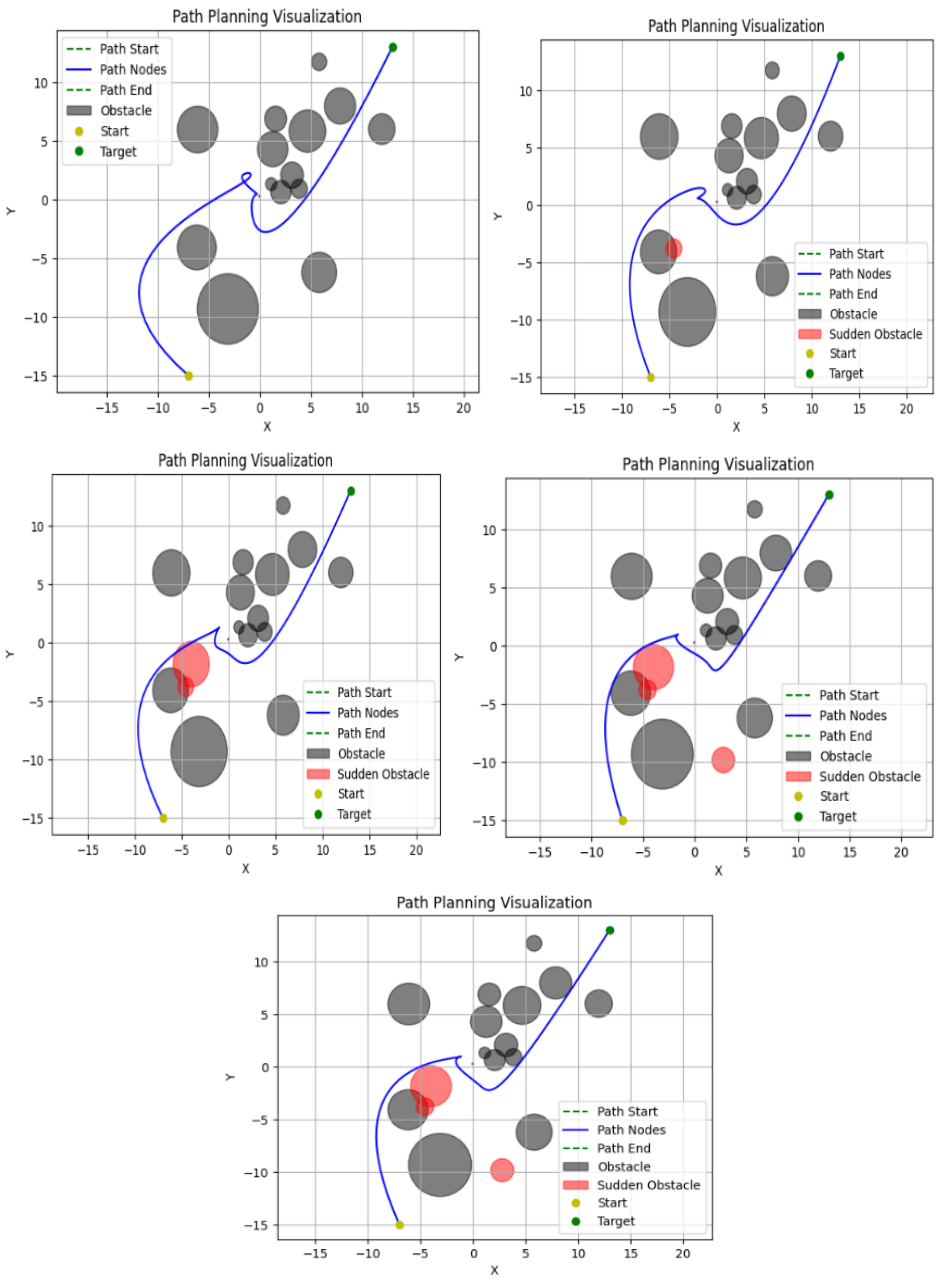


Fig. 8. Obstacles introduction at various iterations (10, 40, 70, 100, and 150)

Fig. 8 gives an example of a motion planning problem of an autonomous vehicle moving through a cluttered setting with moving obstacles added at different iterations. In every plot, the intended path is shown by blue lines, the obstacles by black circles and new obstacles by red circles. Fig. 8 (a) shows that at iteration 10, the vehicle begins to navigate through fixed obstacles. New obstacles are introduced with time: in Fig. 8(b) at iteration 40, Fig. 8(c) at iteration 70, Fig. 8(d) at iteration 100, and Fig. 8(e) at iteration 150. These movements make the vehicle dynamically change its path and still avoid collisions and head to the target. The graphs illustrate how the system can re-plan as the environment changes and provide a safe route despite the unforeseen changes in the environment. This capability is essential for autonomous vehicles operating in complex and dynamic settings, highlighting the robustness of the motion planning algorithm.

Thus, the proposed GCQL method calculates the cumulative rewards of different control sequences in the prediction horizon. Here, the first control action of the optimal control sequence in the prediction horizon is selected for tracking autonomous vehicles in their current state. An episode represents one complete training interaction sequence between the agent and the environment; therefore, episode counts are discrete integer values. The graph representing the calculation of cumulative rewards in various episodes is illustrated in Fig. 9:

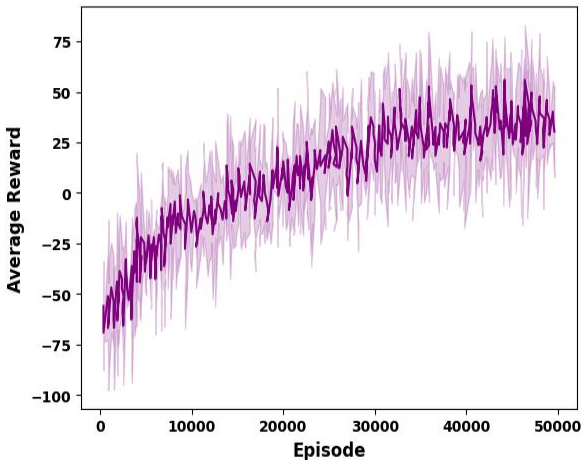


Fig. 9. Estimation of average reward for varying episodes

Fig. 9 represents the cumulative rewards achieved over 50,000 episodes, illustrating the performance of a motion-planning algorithm for autonomous vehicles. The x-axis represents episodes (iterations of learning) and the y-axis represents the average rewards with a higher value representing better decision-making. The dark purple line is the average reward and the shaded area is variability between episodes. The positive trend shows that the algorithm is learning and gaining better and more efficient skills to organize efficient and collision-free paths over time. Fig. 10 shows the functionality of the different path-planning algorithms used in autonomous vehicles to maneuver around the obstacles in an attempt to move out of a specified start point to a specified target point. The black circles are the obstacles, which are static and the red circle is the sudden obstacle that appears dynamically. As depicted in the graph, the suggested approach seems to effectively prevent barriers, including the unexpected ones, and follow a straight and smooth route to the destination, which is more adaptable and plan-oriented. Other algorithms exhibit different obstacle avoidance and path optimization, with some producing slower or less efficient paths.

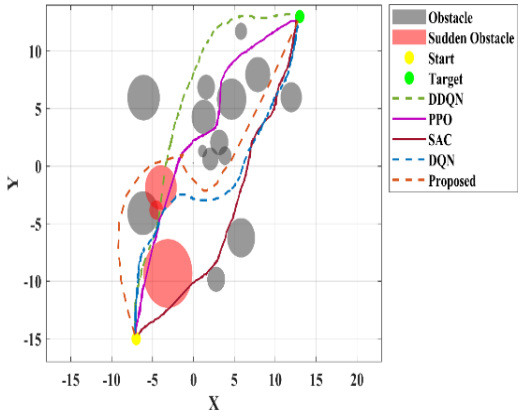


Fig. 10. Path Planning by adding non-obstacles on the way abruptly in the comparison of the current methods

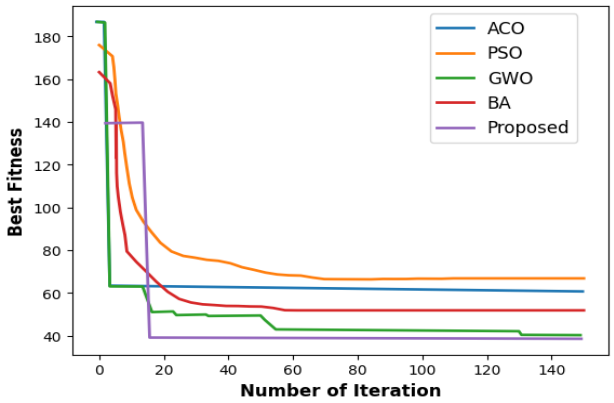


Fig. 11. Comparison with iteration of the best fitness of different optimization algorithms

The performance of different optimization algorithms in convergence in terms of best fitness against iteration is compared in Fig. 11. The suggested technique attains optimal fitness of about 40 in 20 cycles, which is quicker and more efficient in converging. GWO and BA have a lower convergence rate but stabilize at the fitness values of about 50 and 55 respectively. PSO approaches a point of stabilization of about 65 with 40 iterations whereas ACO has the slowest convergence and is at 80. Therefore, in general, the suggested approach is more efficient and accurate than others.

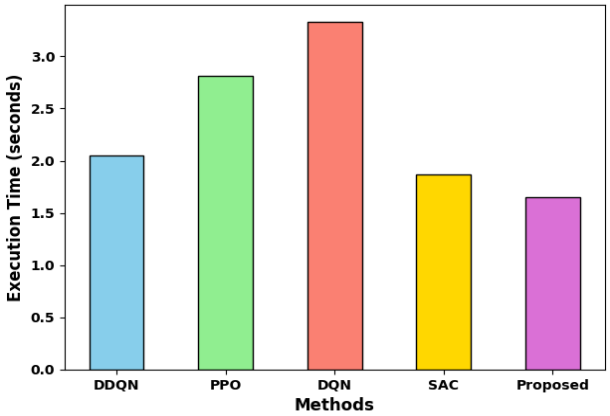


Fig. 12. Comparison of the existing methods of execution time

The execution time of the various methods has been compared in Fig. 12. The longest execution time of 3.33 seconds belongs to DQN, then it is PPO (2.81 seconds) and DDQN (2.05 seconds). SAC is more efficient at approximately 1.87 seconds whereas the proposed method is the lowest with an execution time of approximately 1.65 seconds. The results highlight the proposed method's computational efficiency compared to other approaches. It is necessary to mention that SAC is involved in a continuous action space, and the given GC-PQL framework is using a discretized supervisory control strategy. The comparison measures the overall performance in terms of system level tracking and computational efficiency under the same simulation constraints as opposed to imposing the same control parameterization.

Also, it should be noted that both PPO and SAC are model-free, end-to-end reinforcement learning algorithms that do not require any explicit access to global map information but learn policies by operating in the environment. The suggested framework, on the other hand, embraces a hierarchical structure whereby the global path planning (HDBHP), smoothing (RLWR-CS) and the local obstacle modulation (TM-APF) offer a systematic guidance preceding the tracking control through reinforcement learning. Hence, the suggested strategy has a competitive advantage in the form of previous environmental experience and modular planning format unattainable by independent PPO or SAC agents. The comparison is also made to show the differences in performance in the same simulation conditions; transparency though should acknowledge the differences in the architecture between generic model-free RL, and the proposed model-injected hierarchical system.

Then, the Cumulative Reward of various algorithms in different scenarios is depicted below in the following figures:

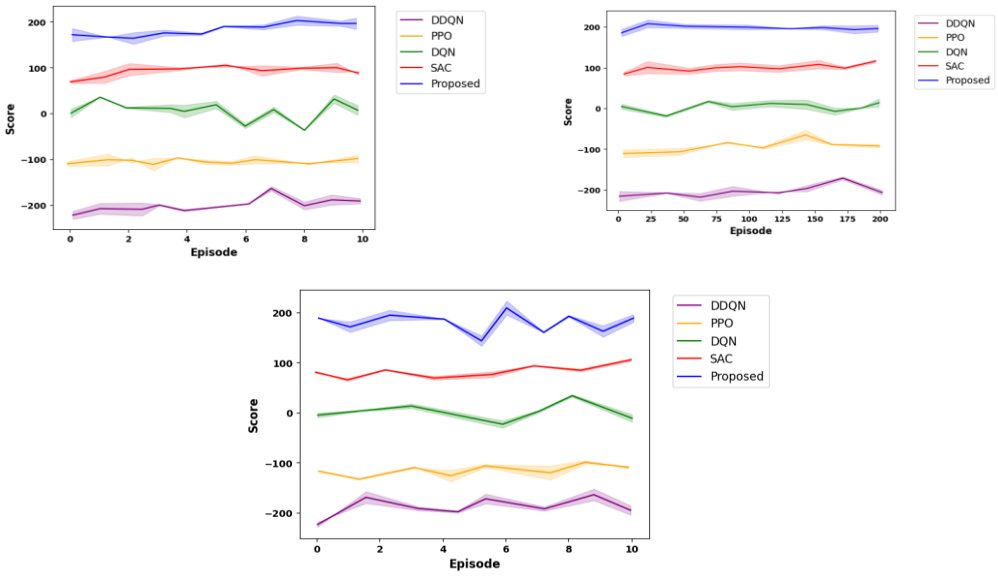


Fig. 13. Comparison of average episodic returns across training episodes for different RL algorithms

In Fig. 13, a higher Cumulative Reward value indicates a better comfort performance with stable velocity while moving. Thus, the Cumulative Rewards at various episodes for different methods are presented in Table 3.

Table 6. Comparison of cumulative rewards upon varying episode counts

Methods	Episode	Score	Episode	Score	Episode	Score	Episode	Score	Episode	Score
DDQN	0.101	-221.8	1.03	-207.7	2.47	-209.0	3.9	-211.5	9.91	-190.9
PPO	-0.081	-109.9	1.29	-100.9	2.10	-102.2	3.6	-97.10	-98.39	9.02
DQN	0.04	0.64	1.03	35.3	1.92	12.21	3.9	4.50	9.02	31.51
SAC	0.02	68.8	1.17	79.1	2.20	95.82	5.3	104.8	88.1	9.25
Proposed	0.081	171.7	1.29	166.5	2.16	163.9	4.4	172.9	9.25	196.1

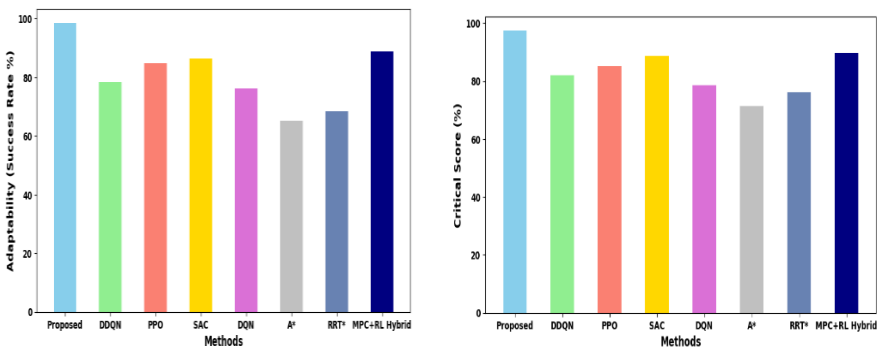
Table 6 illustrates the performance difference of different reinforcement learning algorithms, such as DDQN, PPO, DQN, and SAC, and a suggested one across different episode numbers in terms of scores and numbers of episodes. Scores for DDQN are better with episodes 1.032 at 221.865 episodes and 3.907 at 211.576. PPO is not performing well with scores mostly negative, e.g., -97.106 at 3.684 episodes. DQN and SAC have similar patterns, but SAC has better consistency with 104.82 at 5.324 episodes. The proposed algorithm demonstrates superior learning performance, achieving an average episodic return of 171.7 at episode 968 and converging to 196.1 by episode 9,251. This consistent increase in cumulative reward indicates stable policy convergence and effective long-term learning. Compared to SAC (104.8 at convergence), the proposed method achieves an approximate 87% improvement in final episodic return, demonstrating stronger trajectory tracking and decision-making capability.

5.6. Quantitative Evaluation of Path Planning and Control Methods

The results of the comparison of the performance are presented in Table 7, which represents the excellence of the proposed hierarchical technique in all the critical parameters. The Proposed Method has the highest Cumulative Reward (97.4%), and it exhibits its excellence in terms of accuracy in trajectory, success in tasks and stability generally. Also, its execution time is the lowest (1.65 s), which proves its feasibility in real time, significantly superior to computation-intensive methods. On path smoothness, in terms of jerk values, the suggested approach has a very small value of 0.09. The suggested solution achieves a safety margin, with an average distance of 1.2 meters between the vehicle and obstacles. Another area in which the proposed approach is superior is in the area of adaptability, having a 98.5% success rate when addressing dynamic obstacles and changing scenarios. Generally, the suggested framework is not only more accurate and efficient but also safer, smoother, and more adaptive in path tracking, which is highly appropriate in real-life autonomous driving conditions. The graphical comparison is shown in Fig. 14.

Table 7. Performance comparison of path planning and tracking control techniques

Method	Cumulative Reward (%)	Execution Time (s)	Path Smoothness (Jerk)	Safety Margin (m)	Adaptability (Success Rate %)
Proposed	**97.4**	**1.65**	**0.09**	**1.2**	**98.5**
DDQN	82.1	3.5	0.24	0.78	78.3
PPO	85.3	7.3	0.19	0.89	84.7
SAC	88.7	9.5	0.21	0.93	86.5
DQN	78.5	8	0.27	0.7	76.1
A*	71.4	6	0.35	0.65	65.2
RRT*	76.2	12.34	0.29	0.7	68.4
MPC + RL Hybrid	89.6	14	0.13	1.05	88.9



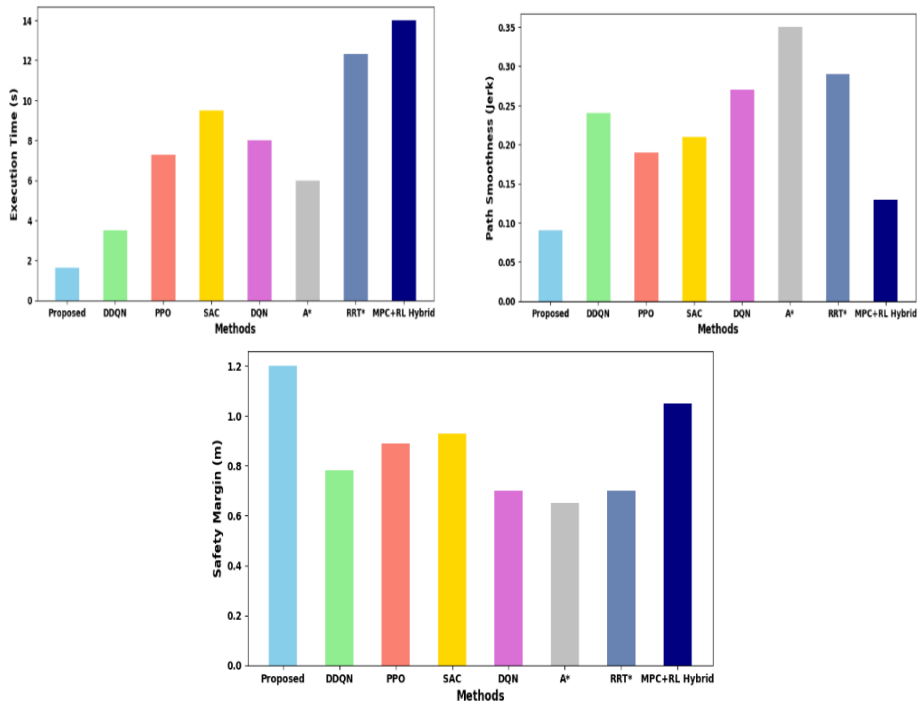


Fig. 14. Graphical results of Path Planning and Tracking Control Techniques comparison

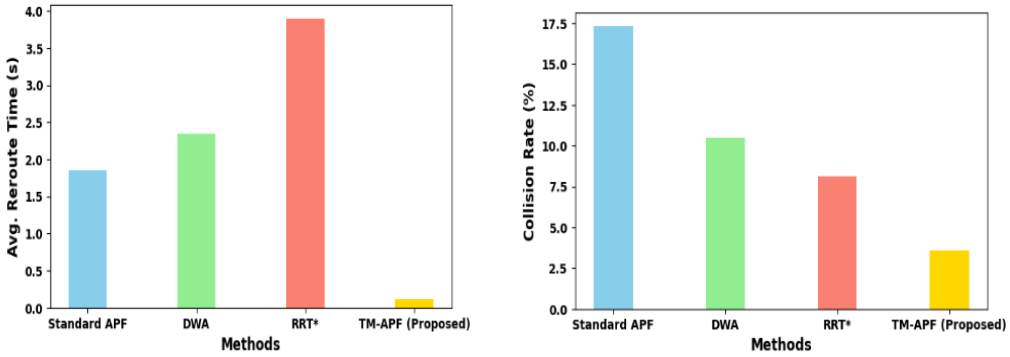
5.7. Performance Comparison of TM-APF vs Existing Techniques

Table 8 results quantitatively demonstrate the impact of the Tent Map-APF (TM-APF) approach compared to other existing techniques like Standard APF, DWA (Dynamic Window Approach), and RRT* (Rapidly-Exploring Random Tree Star).

Table 8. Performance comparison of TM-APF vs existing techniques

Method	Avg. Reroute Time (s)	Collision Rate (%)	Trajectory Smoothness	Success Rate (%)
Standard APF	1.85	17.3	0.124	83.1
DWA	2.34	10.5	0.107	87.4
RRT*	3.89	8.1	0.146	91.5
TM-APF (Proposed)	0.12	3.6	0.063	98.7

Table 8 indicates that the developed TM-APF strategy is superior to the rest in every key performance indicator. It records the shortest average reroute time (0.12s), providing much lower response delay than Standard APF (1.85s), DWA (2.34s), and RRT* (3.89s). TM-APF also boasts the lowest collision ratio (3.6%) and highest smoothness of trajectory (0.063), reflecting improved safety and stable navigation. Most importantly, it has the highest success rate (98.7%), which validates its high adaptability and effective locomotion in real time in dynamic or crowded environments with a stronger level of reliability as compared to traditional methods. Moreover, the graphical representation is shown in Fig. 15.



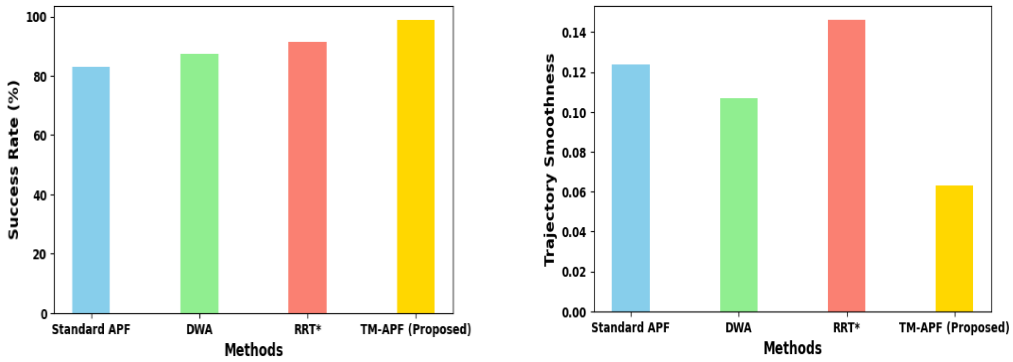


Fig. 15. Graphical results of the TM-APF vs existing techniques

Table 9. Sensitivity analysis of Tent Map chaotic parameters on path security

Chaotic Parameter ($\bar{\mu}$)	Path Smoothness (Curvature Std. Dev.)	Minimum Safety Margin (m)	Oscillation Frequency (Hz)
1.20	0.18	0.92	0.8
1.40	0.22	0.88	1.1
1.60	0.28	0.80	1.7
1.80	0.35	0.74	2.4
2.00	0.40	0.70	3.1

The sensitivity analysis of the chaotic parameters of the Tent Map is detailed in Table 9. Path Smoothness The standard deviation of the curvature of the path value (the less the value, the smoother the path). Minimum Safety Margin is the nearest distance to an obstacle (the further the better). Oscillation Frequency- The frequency at which the direction changes abruptly (the lower the more stable). The findings show that higher values of chaotic parameters (= 1.20–1.40) yield smoother and more stable trajectories with a sufficient safety margin. The greater the chaotic parameter, the smoother and less oscillatory the path and the lower the safety margin and the stability. It is determined that the chaotic coefficient of the Tent Map gives the highest operating range of = 1.20–1.40, which is a trade-off between adaptability and path security.

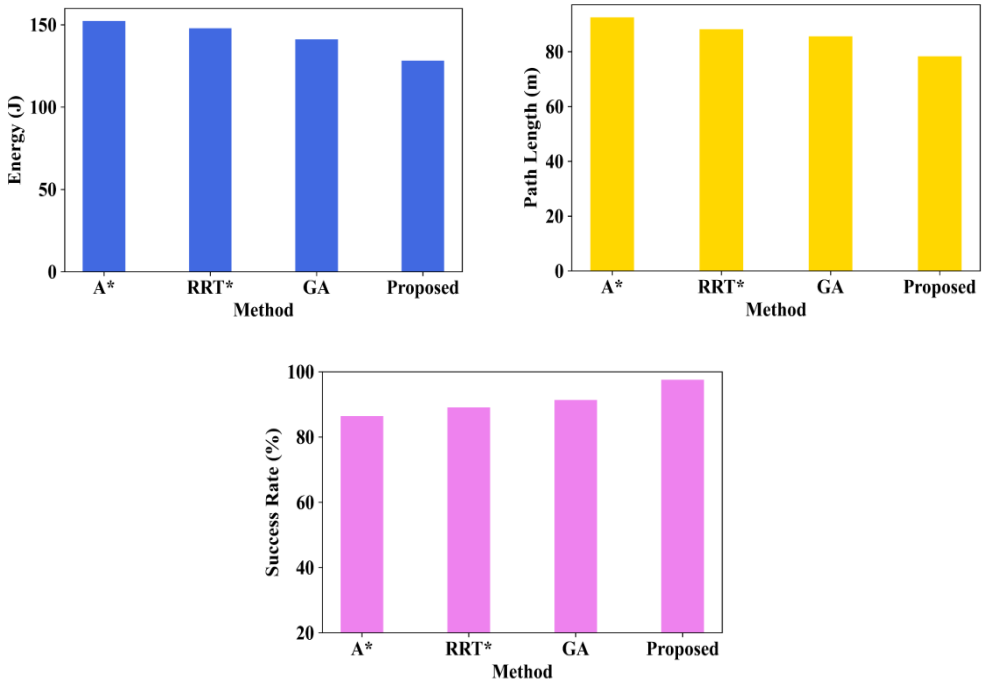


Fig. 16. Comparative performance analysis of path planning methods, a) Energy consumption, b) Path length and c) Success rate comparison

Fig. 16(a-c) provides a detailed comparison of the proposed HDBHP-based hierarchical framework with A*, RRT*, and GA with regard to the energy consumption, path length and the success rate. The proposed method has the lowest energy consumption as indicated in Fig. 16(a) of 128 J compared to 152 J in A*, 148 J in RRT*, and 141 J in GA, which indicates better energy efficiency. As in Fig. 16(b), the proposed method obtains the shortest length of the path; A*, RRT*, and GA yield 92 m, 88 m, and 85 m, respectively, which is a confirmation of improved global optimality. Moreover, Fig. 16(c) indicates that the proposed framework achieves the best success rate of 97% which is better than A* (86%), RRT* (89%), and GA (91%) in the case of dynamic obstacles. All of this confirms that the chosen hybrid optimization and reinforcement learning-based architecture can attain better performance in terms of path distance minimization and energy consumption but still retains strong and stable navigation.

5.8. Discussion

Additionally, we performed benchmark tests for a variety of representative scenarios (simple, cluttered, and dynamic obstacle settings). All the algorithms were benchmarked under equal conditions using equivalent initial configurations and assessment metrics (e.g., path length, total energy, computation time, success rate). The results appear in Table 10.

Table 10. Performance comparison between DBO, HOA, and proposed HDBHP algorithm

Metric	DBO	HOA	HDBHP (Proposed)
Avg. Path Length (m)	82.4	85.1	**78.3**
Energy Consumption (J)	142.6	135.9	**128.2**
Success Rate (%)	91.5	89.8	**97.6**
Avg Computation Time (s)	4.85	4.12	**4.37**
Obstacle Avoidance Score	86.1	88.7	**93.4**
Path Smoothness Index	0.72	0.78	**0.84**

Table 11 reveals that the ablation experiment shows the critical roles of every module in the overall system performance. Full System (All Modules) has the highest performance, Cumulative Reward of 97.4, minimum execution time (1.65 s), smooth path (jerk = 0.09), and maximum adaptability (98.5% success rate).

Table 11. Comprehensive ablation study evaluating the contribution of each major module

Configuration	Cumulative Reward	Execution Time	Path Smoothness	Safety Margin	Adaptability
Full System (All modules)	**97.4**	**1.65**	**0.09**	**1.2**	**98.5**
Without HDBHP	89.8	1.61	0.11	0.95	92.1
Without RLWR-CS	90.3	1.62	0.21	1.15	94.4
Without TM-APF	88.9	1.6	0.1	0.82	89.2
Without GCQL/PQL	86.7	1.9	0.13	1.1	85.6

Upon removal of HDBHP, the Cumulative reward falls drastically by 7.6%, showing the significance of the path-planning algorithm in finding a balance between efficiency and avoidance of obstacles. Without RLWR-CS, the Cumulative Reward falls by 7.1%, with minor degradation in path smoothness and adaptability, highlighting the function of this method in providing smooth, kinematic-constrained paths. Removing TM-APF causes the most significant loss of adaptability (-9.3%), Cumulative Reward (-8.5%), indicating its relevance to dynamic and real-time obstacle avoidance. Finally, the removal of GCQL/PQL leads to the most significant decrease in Cumulative Reward (-10.7%) and adaptability (-12.9%), which illustrates the importance of such approaches to helping to be goal-oriented and make decisions that are context-specific. In summary, the research highlights that all modules are essential in maximizing the performance of the system, and their complete integration is essential for realizing the maximum autonomous driving feature.

Fig. 17 and Table 12 show the contribution of each module of the suggested hierarchical structure separately. The overall performance of the full model is optimal, having a path length of 78.3 m, collision rate of 0.04, jerk of 0.09 m/s^3, and the success rate of 98.5. Disabling the RLWR-CS smoothing module raises jerk to 0.18 m/s^3 and slightly compromises the path length and safety, which underscores its contribution to smoothness in the trajectory and comfort to the passengers. Cutting the TM-APF dynamic avoidance component to half increases the collision rate to 0.12 and reduces the success rate to 91.3 which proves its significance in real-time obstacle management. The adaptive GCQL-PQL control mechanism is required, as without it, the length of the paths and the collision rate further rise, and the success rate reduces to 89.5%, which proves the need to employ reinforcement learning-based adaptive decision-making. Simplified optimizer-only configuration has the worst values in all metrics, the longest path length, the highest collision rate, the highest jerk, and the lowest success rate. On the whole, the research validates that all the components are significant contributors to the performance of the system and support the integrated architecture as opposed to unnecessary complexity.

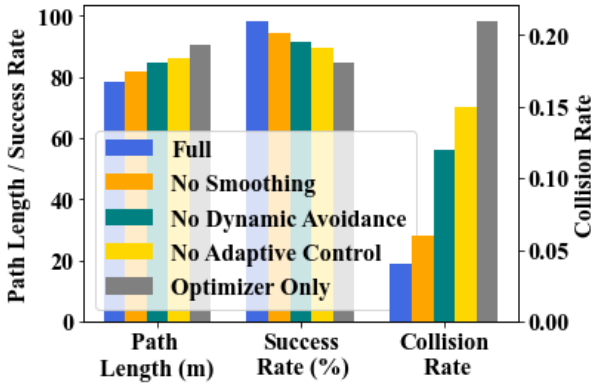


Fig. 17. Navigation-level ablation analysis

Table 12. Navigation-level ablation study

Configuration	Path Length (m)	Collision Rate	Jerk (m/s³)	Success Rate (%)
Full Framework	78.3	0.04	0.09	98.5
Without RLWR-CS (no smoothing)	81.6	0.06	0.18	94.2
Without TM-APF (no dynamic avoidance)	84.9	0.12	0.11	91.3
Without GCQL-PQL (no adaptive control)	86.4	0.15	0.14	89.5
Simplified Optimizer Only	90.7	0.21	0.22	84.6

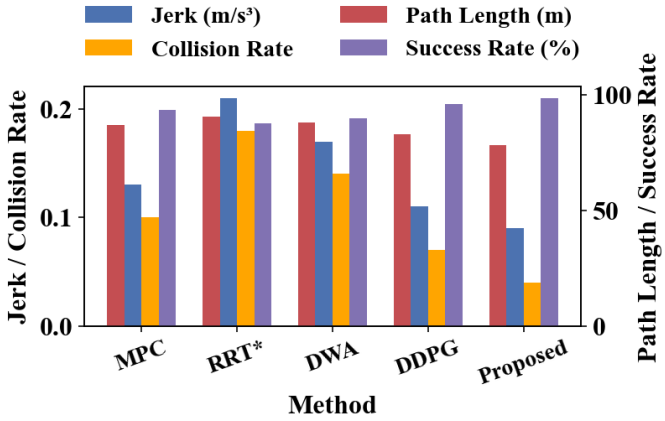


Fig. 18. Graphical comparison of path length, collision rate, jerk, and success rate among MPC, RRT, DWA, DDPG, and the proposed method

Table 13. Quantitative performance comparison of the proposed method with classical and learning-based path planning approaches

Method	Path Length (m)	Collision Rate	Jerk (m/s³)	Success Rate (%)
MPC	86.8	0.10	0.13	93.2
RRT*	90.5	0.18	0.21	87.4
DWA	88.1	0.14	0.17	89.6
DDPG	82.7	0.07	0.11	95.8
Proposed Method	78.3	0.04	0.09	98.5

Fig. 18 shows the graphical comparison of path length, collision rate, jerk, and success rate among MPC, RRT, DWA, DDPG, and the proposed method. Table 13 provides a quantitative validation of the suggested hierarchical approach to the representative classical, sampling-based, and learning-based strategies, including MPC, RRT, DWA, and DDPG, with the most crucial performance indicators being the path length, collision rate, jerk, and the success rate. The suggested approach has the shortest path length of 78.3 m, which is more global and efficient in terms of path generation. It also has the lowest rate of collision at 0.04 which indicates superior safety performance at dynamic conditions. Regarding the smoothness of the motion, the proposed approach provides the lowest value of jerk at 0.09 m/s³, which indicates better comfort of the passengers and consistent control. Moreover, its success rate of 98.5% is the highest, which proves that this device is robust and flexible in complicated navigation conditions. All in all, the evidence shows a definite superiority of the proposed framework in comparison to the traditional planning methods as well as the control approaches

based on reinforcement learning in all the key assessment parameters.

5.9. Theoretical and Empirical Justification of Convergence with PER

In Q-learning, convergence to an optimal policy is guaranteed under exploration conditions like enough exploration, learning rate decay, and sampling without bias. Prioritized Experience Replay (PER) enhances sample efficiency by giving higher priority to experiences with large TD errors and learning more from more informative experiences. But this brings bias to the sampling process, which has to be eliminated to maintain convergence guarantees.

Additionally, PER ensures sufficient exploration through the retention of a small positive lower-bound priority for all transitions, ensuring that each experience has a non-zero probability of being sampled (guaranteeing the infinite visitation condition for convergence). Two significant methods confirm empirically the convergence in the presented system:

Stability of the Q-matrix: Successive Q-matrices are compared using the generalized correlation coefficient (Eq. 49). When this coefficient approaches 1 at all times, we state that convergence is achieved, and this means that Q-values have converged after training.

Policy performance measurements: Measures of accumulated reward average, collision rate and discrepancy of path are monitored during episodes. A gradually increasing reward curve, and plateau or diminishing error rates, are used as the empirical evidence of the process of converging to the best policy. The combination of these analytical and performance-based indicators is what guarantees that the better Q-learning algorithm with PER converges consistently to a robust and optimal control policy in the path-tracking case of the autonomous vehicle.

5.10. Analysis of Chaotic Parameters and Path Security in TM-APF

Although the Tent Map-based Artificial Potential Field (TM-APF) approach introduces variety and flexibility to the dynamic obstacle avoidance by introducing chaotic dynamics, particular consideration should be given to the influence of the chaotic parameters on the path security and the system reliability in general.

Impact of Chaotic Parameters on Path Security: The chaotic manner caused by the Tent Map can help the vehicle to get out of local minima and be more responsive to dynamic obstacles. However, when the chaotic parameters are improperly tuned, they can also cause abrupt trajectories, oscillatory behavior, or random deviations of the path. This instability may reduce smoothness of paths, decrease safety margins, and lead to unsafe maneuvering, especially in practical scenarios where sensor noise or dynamic environments occur.

Addressing Path Instability and Local Minima: To ensure that the advantages of TM-APF do not affect the reliability, we conducted a parameter sensitivity analysis of chaotic coefficients of the Tent Map. By systematic variation and evaluation, safe operating ranges that trade off adaptability and path stability are identified. Path smoothness, minimum safety margin, and oscillation frequency were measured as metrics to gauge the effect on path security. In addition, the introduced framework has a hybrid safeguard mechanism: In case of excessive path instability (e.g., recurrent oscillations or breach of safety margins), the system can downscale adaptively the impact of chaotic dynamics or transition to a more conservative local planner. The RLWR-CS smoothing module further reduces sudden changes by optimizing the planned trajectory so that it becomes kinematically feasible and passenger-friendly.

5.11. Advantages of the Proposed Model

The suggested hierarchical path planning and tracking control system possesses various important benefits. A new Hybrid HDBHP algorithm is used to create the global path when the bio-inspired approaches have the global exploration capability to optimize the path length, obstacle avoidance, and energy consumption. To provide greater flexibility in dynamic conditions, the TM-APF strategy employs chaotic dynamics, which enables the vehicle to get out of local minima and quickly respond to unexpected obstacles, thereby providing safety and a wide range of operations. The RLWR-CS algorithm still optimizes the route to follow vehicle kinematic constraints giving the smoothest routes with minimal jerky actions and maximal passenger comfort. Tracking control is supported by the combination of GCQL and PQL, which facilitates goal-oriented, context-aware decision-making. Prioritized experience replay accelerates convergence, while GCQL keeps attention on navigation goals even in dynamic and uncertain environments. Hierarchical architecture enables effective communication between planning and control layers with real-time updates and continuous execution. Its lightweight nature guarantees realistic deployment, solving the computational limitations of traditional MPC-RL hybrids and the responsiveness problems of offline models. The simulation outcomes confirm the superiority of the method with higher critical performance scores and optimum execution as compared to baseline approaches such as DDQN, PPO, DQN, and SAC.

6. Conclusions

This hierarchical structure shows a highly reliable solution for the AV path planning and tracking control problem by incorporating hybrid optimization, adaptive potential fields, and reinforcement learning. The proposed model offers real-time obstacle avoidance, energy-efficient path planning, and dynamic environment adaptation. The application of HDBHP for global path optimization, RLWR-CS for smoothing, TM-APF for local adjustments, and GCQL with PQL for adaptive tracking guarantees that the system is reactive, safe, and able to handle various driving scenarios. This work is highly relevant to real-world AV deployment because it concentrates on the three key aspects of passenger comfort, safety, and accurate path following to improve AV technology. Simulation results demonstrate that the proposed approach achieves improved Cumulative rewards of 171.704 at episode 968 and 196.141 at episode 9,251, confirming stable reinforcement learning convergence. Furthermore, the given approach has a lower execution time of 1.65 seconds and an outstanding performance on the same simulation conditions in comparison with the baseline methods, including DDQN, PPO, DQN, and SAC. It is worth pointing out that both PPO and SAC are generic model-free RL algorithms, but the proposed framework combines global planning and structured control modules into a hierarchical framework. Therefore, the performance improvements observed are indicative of the benefits of joint optimization planning with reinforcement learning-based tracking as opposed to a one-to-one comparison by using only pure end-to-end RL strategies.

All the Declarations and Statements

Author Contributions Statement

Lakshmi Narayana: Conceptualization, Methodology, and Supervision: Proposed research ideas, Constructed the overall framework, and supervised project execution, Data Curation and Software Implementation: Handled data acquisition, dataset preprocessing, and implementing the research model, Model Training, Validation, and Performance Evaluation: Led the model training process, validated results using standard metrics, and benchmarked performance against existing methods, Formal Analysis, Visualization, and Statistical Analysis: Performed in-depth analysis of experimental results, prepared performance charts, and ensured the statistical robustness of the evaluation.
T. M. N. Vamsi: Writing – Drafted the initial manuscript, contributed to the literature survey, and documented the technical background of the study, Writing – Review and Editing, and Project Management: Reviewed and edited the manuscript, ensured clarity and coherence, and helped coordinate project milestones and deadlines.

All authors have read and agreed to the published version of the manuscript.

Conflict of Interest Statement

The authors declare no conflicts of interest.

Funding Declaration

No funding is provided for the preparation of the manuscript.

Data Availability Statement

Data sharing not applicable to this article as no datasets were generated or analyzed during the current study.

Ethical Declarations

This article does not contain any studies with human participants or animals performed by any of the authors.

Acknowledgments

We sincerely thank the experts for their professional evaluation and valuable recommendations, which have contributed to improving the quality of the experiment and the reliability of its results.

Declaration of Generative AI in Scholarly Writing

The authors declare that generative artificial intelligence (AI) tools were used only for language editing and improving the clarity of the manuscript. The AI tools did not contribute to the research design, data collection, data analysis, interpretation of results, or the development of the scientific content. The authors take full responsibility for the accuracy, originality, and integrity of the manuscript.

Abbreviations

The following abbreviations are used in this manuscript:

HDBHP- Hybrid Dung Beetle and Hippopotamus Optimization
RLWR-CS- Robust Locally Weighted Regression with Curvature Smoothing

TM-APF- Tent Map-Artificial Potential Field
RL- Reinforcement Learning
PQL- Prioritized Q-learning
IVs- Intelligent Vehicles
AV- Autonomous Vehicle
MPC- Model Predictive Control
DBO- Dung Beetle Optimization
HPO- Hippopotamus Optimization
GHPF- Global Heuristic Potential Field
GC-PQL- Goal-Conditional Prioritized Q-learning

References

[1] Teng, Siyu, Xuemin Hu, Peng Deng, Bai Li, Yuchen Li, Yunfeng Ai, "Motion planning for autonomous driving: The state of the art and future perspectives", IEEE Transactions on Intelligent Vehicles. Vol.8, No.6, pp.3692–3711, 2023. DOI: 10.1109/TIV.2023.3274536

[2] Yan, Yongjun, Lin Peng, Jinxiang Wang, Hui Zhang, Tong Shen, and Guodong Yin. "A hierarchical motion planning system for driving in changing environments: Framework, algorithms, and verifications." IEEE/ASME Transactions on Mechatronics 28, no. 3 (2022): 1303-1314. "A hierarchical motion planning system for driving in changing environments: Framework, algorithms, and verifications", IEEE/ASME Trans Mechatron. Vol.28, No.3, pp.1303–1314, 2022. DOI: 10.1109/TMECH.2022.3219617

[3] He, Xiangkun, Haohan Yang, Zhongxu Hu, and Chen Lv. "Robust lane change decision making for autonomous vehicles: An observation adversarial reinforcement learning approach", IEEE Transactions on Intelligent Vehicles. Vol.8, No.1, pp.184–193, 2022. DOI: 10.1109/TIV.2022.3165178

[4] Pérez-Gil, Óscar, Rafael Barea, Elena López-Guillén, Luis M. Bergasa, Carlos Gomez-Huelamo, Rodrigo Gutiérrez, and Alejandro Diaz-Diaz, "Deep reinforcement learning based control for autonomous vehicles in CARLA", Multimedia Tools and Applications. Vol.81, No.3, pp.3553–3576, 2022. DOI: 10.1007/s11042-021-11437-3

[5] He, Yixu, Yang Liu, Lan Yang, and Xiaobo Qu, "Deep adaptive control: Deep reinforcement learning-based adaptive vehicle trajectory control algorithms for different risk levels", IEEE Transactions on Intelligent Vehicles. Vol.9, No.1, pp.1654–1666, 2023. DOI: 10.1109/TIV.2023.3303408

[6] Zhang, Yuxiang, Xiaoling Liang, Dongyu Li, Shuzhi Sam Ge, Bingzhao Gao, Hong Chen, and Tong Heng Lee, "Adaptive safe reinforcement learning with full-state constraints and constrained adaptation for autonomous vehicles", IEEE Transactions on Cybernetics. Vol.54, No.3, pp.1907–1920, 2023. DOI: 10.1109/TCYB.2023.3283771

[7] Tang, Xiaolin, Kai Yang, Hong Wang, Wenhao Yu, Xin Yang, Teng Liu, and Jun Li, "Driving environment uncertainty-aware motion planning for autonomous vehicles", Chinese Journal of Mechanical Engineering, Vol.35, No.1, pp.120, 2022. DOI: 10.1186/s10033-022-00790-5

[8] Kim, Dongchan, Gihoon Kim, Hayoung Kim, and Kunsoo Huh, "A hierarchical motion planning framework for autonomous driving in structured highway environments", IEEE Access. Vol.10, pp.20102–20117, 2022. DOI: 10.1109/ACCESS.2022.3152187

[9] Zhang, Yong, Kangting Liu, Feng Gao, and Fengkui Zhao, "Research on path planning and path tracking control of autonomous vehicles based on improved APF and SMC", Sensors 23, Vol.23, No.18, pp.7918, 2023. DOI: 10.3390/s23187918

[10] Du, Guodong, Yuan Zou, Xudong Zhang, Zirui Li, and Qi Liu, "Hierarchical motion planning and tracking for autonomous vehicles using global heuristic based potential field and reinforcement learning based predictive control", IEEE Transactions on Intelligent Transportation Systems 24, Vol.24, No.8, pp.8304–8323, 2023. DOI: 10.1109/TITS.2023.3266195

[11] Zhang, Xiao, Tong Zhu, Lei Du, Yueqi Hu, and Haoxue Liu, "Local path planning of autonomous vehicle based on an improved heuristic bi-RRT algorithm in dynamic obstacle avoidance environment", Sensors 22, Vol.22, No.20, pp.7968, 2022. DOI: 10.3390/s22207968

[12] Zhang, Xinglong, Wenxin Zhang, Youqun Zhao, Hai Wang, Fen Lin, and Yingfeng Cai, "Personalized motion planning and tracking control for autonomous vehicles obstacle avoidance", IEEE Transactions on Vehicular Technology, Vol.71, No.5, pp.4733–4747, 2022. DOI: 10.1109/TVT.2022.3152542

[13] Almazrouei, Khawla, Ibrahim Kamel, and Tamer Rabie, "Dynamic obstacle avoidance and path planning through reinforcement learning", Applied Sciences, Vol.13, No.14, pp.8174, 2023. DOI: 10.3390/app13148174

[14] Huang, Zhiyu, Jingda Wu, and Chen Lv, "Efficient deep reinforcement learning with imitative expert priors for autonomous driving", IEEE Transactions on Neural Networks and Learning Systems, Vol.34, No.10, pp.7391–7403, 2022. DOI: 10.1109/TNNLS.2022.3142822

[15] Bellotti, Francesco, Luca Lazzaroni, Alessio Capello, Marianna Cossu, Alessandro De Gloria, and Riccardo Berta, "Explaining a deep reinforcement learning (DRL)-based automated driving agent in highway simulations", IEEE Access, Vol.11, pp.28522–28550, 2023. DOI: 10.1109/ACCESS.2023.3259544

[16] Li, Guofa, Siyan Lin, Shen Li, and Xingda Qu, "Learning automated driving in complex intersection scenarios based on camera sensors: A deep reinforcement learning approach", IEEE Sensors Journal, Vol.22, No.5, pp.4687–4696, 2022. DOI: 10.1109/JSEN.2022.3146307

[17] Hu, Hui, Yuge Wang, Wenjie Tong, Jiao Zhao, and Yulei Gu, "Path planning for autonomous vehicles in unknown dynamic environment based on deep reinforcement learning", Applied Sciences, Vol.13, No.18, pp.10056, 2023. DOI: 10.3390/app131810056

[18] N.S. Manikandan, Ganesan Kaliyaperumal, and Yong Wang, "Ad hoc-obstacle avoidance-based navigation system using deep reinforcement learning for self-driving vehicles", IEEE Access, Vol.11, pp.92285–92297, 2023. DOI: 10.1109/ACCESS.2023.3297661

[19] Tang, Xiaolin, Guichuan Zhong, Shen Li, Kai Yang, Keqi Shu, Dongpu Cao, and Xianke Lin, "Uncertainty-aware decision-making for autonomous driving at uncontrolled intersections", IEEE Transactions on Intelligent Transportation Systems, Vol.24, No.9, pp.9725–9735, 2023. DOI: 10.1109/TITS.2023.3283019
[20] Zhang, Sunan, Weichao Zhuang, Bingbing Li, Ke Li, Tianyu Xia, and Bo Hu, "Integration of planning and deep reinforcement learning in speed and lane change decision-making for highway autonomous driving", IEEE Transactions on Transportation Electrification, 2024. DOI: 10.1109/TTE.2024.3393293
[21] Yang, Lei, Chao Lu, Guangming Xiong, Yang Xing, and Jianwei Gong, "A hybrid motion planning framework for autonomous driving in mixed traffic flow", Green Energy and Intelligent Transportation, Vol.1, No.3, pp.100022, 2022. DOI: 10.1016/j.geits.2022.100022
[22] Antonio, Guillen-Perez, and Cano Maria-Dolores, "Multi-agent deep reinforcement learning to manage connected autonomous vehicles at tomorrow's intersections", IEEE Transactions on Vehicular Technology,71, No.7, pp.7033–7043, 2022. DOI: 10.1109/TVT.2022.3169907
[23] Li, Guofa, Yifan Yang, Shen Li, Xingda Qu, Nengchao Lyu, and Shengbo Eben Li, "Decision making of autonomous vehicles in lane change scenarios: Deep reinforcement learning approaches with risk awareness", Transportation Research Part C: Emerging Technologies, Vol.134, pp.103452, 2022. DOI: 10.1016/j.trc.2021.103452
[24] Albarella, Nicola, Dario Giuseppe Lui, Alberto Petrillo, and Stefania Santini, "A hybrid deep reinforcement learning and optimal control architecture for autonomous highway driving", Energies. Vol.16, No.8, pp.3490, 2023. DOI: 10.3390/en16083490
[25] Dang, Fengying, Dong Chen, Jun Chen, and Zhaojian Li, "Event-triggered model predictive control with deep reinforcement learning for autonomous driving", IEEE Transactions on Intelligent Vehicles, Vol.9, No.1, pp.459–468, 2023. DOI: 10.1109/TIV.2023.3329785
[26] Cao, Zhong, Diange Yang, Kun Jiang, Shaobing Xu, Sijia Wang, Minghan Zhu, and Zhongyang Xiao, "A geometry-driven car-following distance estimation algorithm robust to road slopes", Transportation Research Part C: Emerging Technologies, Vol.102, pp.274–288, 2019. DOI: 10.1016/j.trc.2019.03.009
[27] Zhang, Bingzhan, Zhiyuan Li, Yaoyao Ni, and Yujie Li, "Research on path planning and tracking control of automatic parking system", World Electric Vehicle Journal, Vol.13, No.1, pp.14, 2022. DOI: 10.3390/wevj13010014
[28] Xue, Jiankai, and Bo Shen, "Dung beetle optimizer: A new meta-heuristic algorithm for global optimization", The Journal of Supercomputing, Vol.79, No.7, pp.7305–7336, 2023. DOI: 10.1007/s11227-022-04959-6
[29] Amiri, Mohammad Hussein, Nastaran Mehrabi Hashjin, Mohsen Montazeri, Seyedali Mirjalili, and Nima Khodadadi, "Hippopotamus optimization algorithm: A novel nature-inspired optimization algorithm", Scientific Reports, Vol.14, No.1, pp.5032, 2024. DOI: 10.1038/s41598-024-54910-3
[30] Ma, Xiaolu, Rui Gong, Yibo Tan, Hong Mei, and Chengcheng Li, "Path planning of mobile robot based on improved PRM based on cubic spline", Wireless Communications and Mobile Computing, Vol.2022, No.1, pp.1632698, 2022. DOI: 10.1155/2022/1632698
[31] Gabr, Mohamed, Hana Younis, Marwa Ibrahim, Sara Alajmy, and Wassim Alexan, "Visual data enciphering via DNA encoding, S-box, and tent mapping", In 2022 IEEE 5th International Conference on Image Processing Applications and Systems (IPAS), pp.1–6, 2022. DOI: 10.1109/IPAS55744.2022.10052832
[32] Xie, Songtao, Junyan Hu, Parijat Bhowmick, Zhengtao Ding, and Farshad Arvin, "Distributed motion planning for safe autonomous vehicle overtaking via artificial potential field", IEEE Transactions on Intelligent Transportation Systems, Vol.23, No.11, pp.21531–21547, 2022. DOI: 10.1109/TITS.2022.3189741
[33] B Eysenbach, Benjamin, Tianjun Zhang, Sergey Levine, and Russ R. Salakhutdinov, "Contrastive learning as goal-conditioned reinforcement learning", Advances in Neural Information Processing Systems, Vol.35, pp.35603–35620, 2022. DOI: 10.48550/arXiv.2206.07568
[34] Qu, Zhiyi, Xue Zhao, Huihui Xu, Hongying Tang, Jiang Wang, and Baoqing Li, "An improved Q-learning-based sensor-scheduling algorithm for multi-target tracking", Sensors 22, Vol.22, No.18, pp.6972, 2022. DOI: 10.3390/s22186972

Authors' Profiles



I. Lakshmi Narayana, currently pursuing his PhD in the department of Computer Science and Systems Engineering, Andhra University, Visakhapatnam. He is working as Assistant Professor in the Department of Information Technology, Seshadri Rao Gudlavalleru Engineering College, Gudlavalleru. His research interests include the Application of machine learning algorithms in Autonomous vehicles, Image segmentation and classification, routing protocols for IoT communication.



Dr. T. M. N. Vamsi is working as Associate Professor in the Department of Computer Science and Engineering at GITAM Deemed to be University, Visakhapatnam, Andhra Pradesh. He received his PhD in Computer Science and Engineering from JNTUH, Hyderabad in the year 2016. He is having 24 years of teaching, research, and administrative experience in various technical higher education Institutions. His research interests are in the development of protocols for the Internet of Things and vehicular networks, Soft Computing and Bioinformatics.

How to cite this paper:

Lakshmi Narayana, T. M. N. Vamsi, "Hybrid Hierarchical Path Planning and Adaptive Tracking Control for Autonomous Vehicles via HDBHP Optimization and Reinforcement Learning", International Journal of Intelligent Systems and Applications (IJISA), Vol.18, No.5, pp.74-108, 2026. DOI: 10.5815/ijisa.2026.05.05