BULLETIN OF THE POLISH ACADEMY OF SCIENCES
TECHNICAL SCIENCES, Vol. 74(3), 2026, Article number: e158296
DOI: 10.24425/bpasts.2026.158296

A hierarchical RRT-DWA planner for autonomous parking in dynamic environments

Hao CHEN[1], Jiateng YANG[2], Xiangmei YE[1], Weiwu CHEN[2], Wenfeng GUO[3*], Yuruo WANG[4], and Gan SHEN[5]

[1] Department of Artificial Intelligence, Zhejiang Business Technology Institute, Ningbo, 315000, China
[2] School of Computer Science, City University of Hong Kong, Hong Kong, 999077, China
[3] School of Vehicle and Mobility, State Key Laboratory of Intelligent Green Vehicle and Mobility, Tsinghua University, Beijing, 100000, China
[4] Pan-Asia Technical Automotive Center Co., Ltd., Shanghai, 201805, China
[5] School of Automotive and Transportation Engineering, Hefei University of Technology, Hefei, 230000, China

Abstract. Aiming to tackle the problems of low adaptability to dynamic environments and low planning efficiency of traditional automatic parking path-planning algorithms, this paper proposes a hierarchical path-planning framework that integrates the improved rapidly-exploring random tree (RRT) algorithm and the dynamic window approach (DWA). Firstly, at the global planning level, a Gaussian-uniform mixed-distribution sampling strategy is adopted to optimize the growth direction of the random tree, and a dynamic step-size mechanism is incorporated to improve the algorithm expansion efficiency. Secondly, the artificial potential field (APF) method is introduced to optimize the RRT-generated path nodes, ensuring the geometric safety clearance for the vehicle chassis. Subsequently, at the local planning level, these optimized nodes serve as waypoints to guide the DWA. Dynamic obstacle-avoidance weight is introduced into the evaluation function of DWA, and this RRT-DWA collaborative framework effectively solves the problems of dynamic obstacle-avoidance and local stagnation. Finally, for the terminal parking maneuver, the Reeds-Shepp (RS) curve is used to smoothly adjust the vehicle pose to match the parking end-point. Finally, a joint simulation is carried out in Carsim/Simulink through the pure-pursuit control algorithm. The simulation experiments show that the maximum tracking error of the planned global path in parallel, perpendicular, and diagonal parking scenarios is within 0.35 m, and the distance from dynamic obstacles is greater than 2 m, which confirms that the planned path is rational.

Keywords: automatic parking; improved RRT; hierarchical path-planning; Reeds-Shepp curve; hybrid algorithm.

1. INTRODUCTION

In recent years, vehicle autonomous driving technology has become a major research direction. As an important part of autonomous driving technology, automatic parking technology has been widely studied. According to the data of the Society of Automotive Engineers (SAE), the market penetration rate of automatic parking systems reached 32% in 2023. Path planning for parking systems has always been an active area of research in automatic parking. Currently, research on parking path planning mainly focuses on two aspects: local path planning and global path planning. The former emphasizes the state of the vehicle in a local position, while the latter pays more attention to the accessibility of the global path. Cai *et al*. [1] proposed path-planning algorithms for parallel parking and perpendicular parking, considering the space of the parking corridor and solving the problem of discontinuous vehicle paths. Li *et al*. [2] generated local parking paths through a hybrid dynamic A* algorithm, enabling parking in narrow parking spaces. Li *et al*. [3] proposed a parallel-parking path-planning method based on a three-stage curve interpolation method. This method achieved parking in narrow positions through curve splicing. Zheng *et al*. [4] proposed an optimal path-planning algorithm based on a combination of circular arcs and straight lines in the presence of predefined obstacles for parking in narrow positions. Yu *et al*. [5] proposed a new parallel parking path-planning method based on the combination of a fifth-order polynomial curve and an improved S-shaped function, which solved the problems of discontinuous curvature and nonreturn of the terminal tires. Although local planners can generate feasible parking trajectories, the overall parking process typically requires awareness of the global environment. This necessity is addressed through global parking path planning.

Global path planning mainly includes graph-search algorithms and sampling-based algorithms. The most common ones are the A* algorithm [6, 7], Dijkstra's algorithm [8, 9], the RRT algorithm [10, 11], the RRT* algorithm [12, 13], the genetic algorithm [14], etc. Currently, many scholars conduct in-depth research on the above-mentioned global path-planning algorithms and propose various optimization methods. For parking path planning, the above-mentioned global path-planning algorithms are usually combined with various other algorithms to meet the kinematic or dynamic constraints of the vehicle. For example, Zheng *et al*. [15] proposed a path-planning method based on the

*e-mail: gwf0330@163.com

Manuscript submitted 2025-11-13, revised 2026-01-18, initially accepted for publication 2026-02-22, published in January 2026.

RRT for vehicle nonholonomic constraints and kinematic models. A kinematic model of the parking lot was established according to the vehicle kinematic equations, followed by a proposal of nonholonomic constraints. Based on this model, the RRT algorithm was used to search for the constrained parking path. To optimize the search efficiency, two strategies, namely target preference and double-RRT, were adopted, and a cost function was added for optimization. Qian *et al*. [16] established a vehicle motion model based on the Ackermann principle, obtained the minimum turning radius, and then constructed a mathematical model for automatic parking. They planned the corresponding path according to arcs and straight lines, and used the asymptotically optimal RRT algorithm to calculate the optimal parking space and the driving trajectory from the current position to the optimal parking space. Manav *et al*. [17] proposed a realistic iterative analysis method of the deterministic parking behavior model (IAM), which was combined with the closed-loop rapidly-exploring random tree (CL-RRT) method for cascade path planning. Jin *et al*. [18] introduced the TargetTree-RRT* algorithm for complex environments and designed a target tree using the Sleeveoid path to address this curvature discontinuity. Finally, a continuous and efficient parking path was obtained.

While static path planning establishes the foundation for autonomous parking, handling dynamic obstacles remains a critical challenge. Traditional approaches for dynamic environments include velocity obstacle (VO) methods [19] and dynamic artificial potential fields (APF) [20]. However, VO-based methods often assume linear obstacle motion, which may be insufficient for complex parking scenarios, while dynamic APF can suffer from local minima issues in changing environments. Recently, optimization-based local planners like the dynamic window approach (DWA) [21, 22] have been extended to dynamic scenarios. However, standard DWA often lacks a predictive risk assessment for moving objects, leading to reactive rather than proactive avoidance. To address this, this paper innovatively introduces a time-to-collision (TTC) based dynamic risk cost function into the DWA planner. This enhancement allows the vehicle to explicitly evaluate the potential collision risk with moving obstacles in the temporal domain, thereby balancing trajectory efficiency with dynamic safety.

To address the limitations of existing methods in dynamic environments, this paper proposes a hierarchical path-planning framework integrating an improved RRT, APF, DWA, and Reeds-Shepp curves. The main contributions are summarized as follows:

1. Global panning optimization: An improved RRT algorithm is proposed, utilizing a Gaussian-uniform mixed sampling strategy and a dynamic step-size mechanism to significantly reduce redundant nodes and improve convergence speed.
2. Dynamic obstacle avoidance: A collaborative RRT-DWA framework is established. A dynamic risk cost function based on time-to-collision (TTC) is introduced into the DWA, enabling the vehicle to proactively avoid moving obstacles while following global guidance.
3. Complete parking solution: The framework effectively integrates global exploration, local dynamic avoidance, and terminal pose adjustment (using RS curves), ensuring collision-free parking in parallel, perpendicular, and diagonal scenarios.

The remainder of this paper is organized as follows: Section 2 establishes the vehicle kinematic model; Section 3 details the improved RRT algorithm; Section 4 presents the fusion algorithm, including APF optimization, improved DWA, and RS curve generation; Section 5 discusses the simulation results, and Section 6 concludes the paper.

2. THE MODEL OF THE VEHICLE

To accurately describe the vehicle motion during parking, a kinematic bicycle model based on the center of mass (CoM) [23] is established, as shown in Fig. 1. Let point C denote the vehicle CoM. The distance from the CoM to the front axle is defined as l_f, and the distance to the rear axle is l_r.

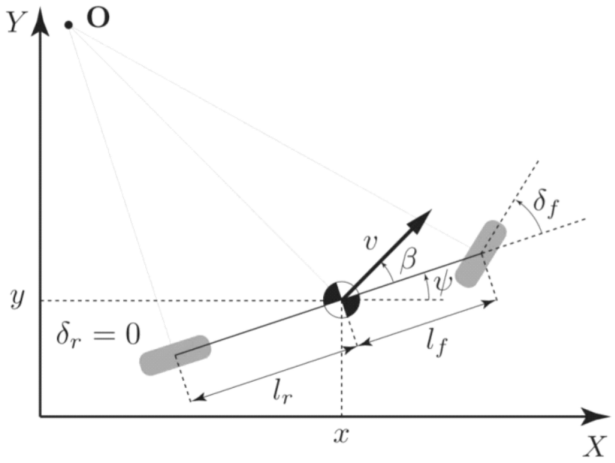


Fig. 1. Two-degree-of-freedom bicycle model

Unlike the simple geometric model based on the rear axle center, the motion of the vehicle CoM considers the sideslip angle β. The kinematic equations are expressed as follows:

$$\begin{aligned} \dot{x} &= v\cos(\psi+\beta), \\ \dot{y} &= v\sin(\psi+\beta), \\ \dot{\psi} &= \frac{v}{l_r}\sin\beta, \\ \dot{v} &= a, \end{aligned} \tag{1}$$

where v is the linear velocity at the CoM, ψ is the heading angle (yaw), and a is the acceleration. The sideslip angle β is determined by the steering angle δ_f and the geometric parameters of the vehicle:

$$\beta(t) = \arctan\left(\frac{l_r}{l_f+l_r}\tan(\delta_f)\right).$$

This model establishes the relationship between the control inputs (velocity v, steering angle δ_f) and the state of the vehicle in the global coordinate system.

3. THE RRT ALGORITHM AND ITS IMPROVEMENT

3.1. Traditional RRT algorithm

The traditional rapidly-exploring random tree (RRT) algorithm [18,24] is a path-planning method based on random sampling, which rapidly explores high-dimensional and complex spaces. However, its reliance on a uniform random sampling strategy means its exploration scope and direction are governed by a uniform distribution. In complex obstacle environments, such as a parking lot, this results in a large amount of redundancy in invalid areas and leads to low planning efficiency [25]. To address this issue, this paper proposes an improved RRT algorithm by introducing a dynamic biased sampling strategy based on a Gaussian-uniform mixture [26] and a dynamic step-size mechanism, aiming to reduce redundant nodes and accelerate algorithm convergence.

3.2. Improvement of the RRT algorithm

The RRT algorithm relies on a random sampling strategy to explore the map space. Its exploration scope and direction are governed by a uniform random distribution, resulting in a large amount of redundancy in invalid areas, especially in complex obstacle environments. To address the redundancy caused by the uniform sampling of the traditional RRT algorithm, a mixture of Gaussian-uniform distribution is introduced, and its probability density function is shown in formula (2):

$$p(q_{\text{rand}}) = \alpha(t) \cdot N\left(q_{\text{goal}}, \sigma(t)\right) + (1 - \alpha(t)) \cdot U(C_{\text{free}}). \quad (2)$$

Among them, the detailed formula $\alpha(t)$ is shown in (3):

$$\alpha(t) = \left[\alpha_{\max} \cdot \left(1 - \frac{d_{\min}(t)}{d_{\text{init}}}\right) + \alpha_{\min}\right] \cdot \partial(t). \quad (3)$$

This value represents the target-bias weight. Here, $\alpha_{\min} \in [0.1, 0.3]$ denotes the minimum target-bias weight, $d_{\min}$ represents the minimum distance from the current tree-node to the target point, and d_{init} represents the shortest distance from the starting point to the target point. $\partial(t)$ is the obstacle-attenuation factor. In the expression $\partial(t) = \gamma^{N_{\text{fail}}(t)}$, $N_{\text{fail}}(t)$ is the number of collisions that occur when the current node expands towards the target point, $\gamma \in (0, 1)$ is the obstacle-attenuation coefficient, and $\alpha_{\max} \in [0.6, 0.9]$ is the maximum target-bias weight. Among these, the formula for $\sigma(t)$ is shown as formula (4) below:

$$\sigma(t) = \left[\sigma_{\max} \cdot \left(1 - \frac{d_{\min}(t)}{d_{\text{init}}}\right)^{\kappa} + \sigma_{\min}\right] (1 + \varpi \cdot N_{\text{fail}}(t)), \quad (4)$$

where $\sigma(t)$ represents the sampling range. In the formula, $\sigma_{\min}$ and $\sigma_{\max}$ are the lower and upper limits of the variance, respectively. The variance contraction factor is $\kappa \in (0, 1)$, and the equation expansion factor is $\varpi \in (0, 1)$. When there are no obstacles, the algorithm will quickly search towards the target point. However, when the algorithm is close to the target point, and there are obstacles near the target point, if the biased sampling collides with the obstacles, the algorithm will expand the sampling range to bypass the obstacles. In the above formula, $N\left(q_{\text{goal}}, \sigma(t)\right)$ represents the Gaussian distribution, and $U(C_{\text{free}})$ represents the uniform distribution. Dynamic sampling is carried out according to the distance between the random tree and the target point, as well as the obstacle information. As shown in Fig. 2.

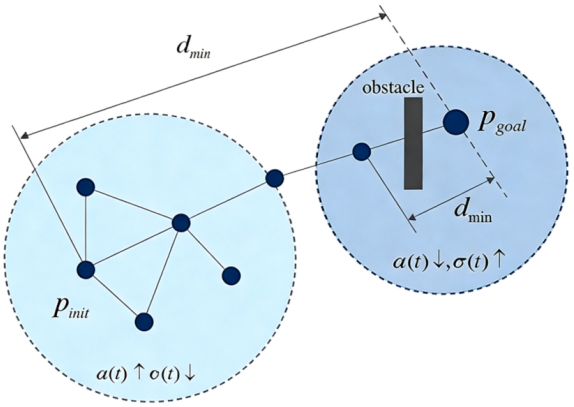


Fig. 2. The situation of biased sampling distribution

The random tree is far from the target point, $d_{\min} \approx d_{\text{init}}$, $\alpha(t) \approx \alpha_{\min}$, and at this time, $\sigma(t) \approx \sigma_{\max}$ is mainly sampled by the uniform distribution. When the distance to the target point becomes increasingly shorter, $d_{\min} \approx 0$, $\alpha(t) \approx \alpha_{\max}$, and at this time, $\sigma(t) \approx \sigma_{\min}$ is mainly in the Gaussian distribution. But if there are obstacles between the target point and the random tree, resulting in the failure of the random tree expansion, the bias weight will be reduced, and the sampling range will be expanded. After the above dynamic sampling optimization, while considering the biased sampling, the influence of obstacles is taken into account, avoiding the problem that the local stagnation of the random tree leads to the failure of obtaining a successful path. At the same time, the sampling range is reduced, redundancy is decreased, and efficiency is improved. Considering the situation of obstacles, when it encounters an obstruction while growing towards the target point, the algorithm will reduce the probability of biased sampling, thus bypassing the obstacles to reach the target point.

After the optimization of dynamic biased sampling, the RRT algorithm significantly reduced redundant nodes. To further improve the adaptability of the algorithm, a dynamic step-size is introduced to accelerate the convergence of the algorithm. The core idea of the dynamic step-size is to determine the growth distance according to the density of the map obstacles. For open environments, the step-size should be increased, while for narrow environments and environments with dense obstacles, the step-size should be appropriately reduced. There are various designs for the dynamic step-size. In this paper, a method of dynamically adjusting the step-size according to the obstacle density is designed. Specifically, as shown in Fig. 3.

In the preset map environment, assume the position of the current random-tree node is $p_{rmnear}(x_i, y_i)$. Then, a square region Ω_i with a length of $2R$ is established centered at this point. In the square region, the grid occupied by the obstacle is represented

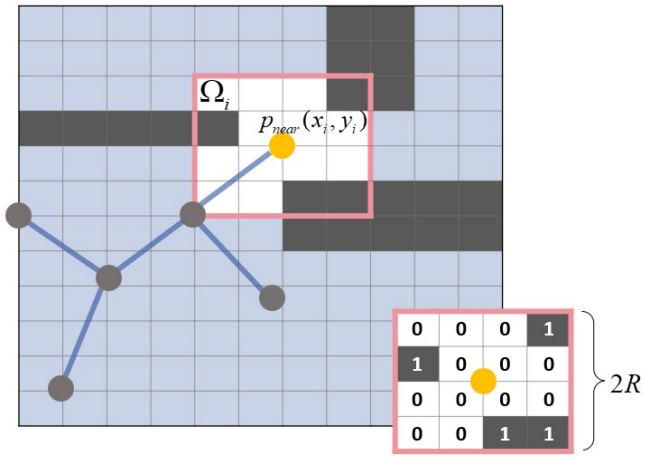


Fig. 3. The node neighborhood in a grid map

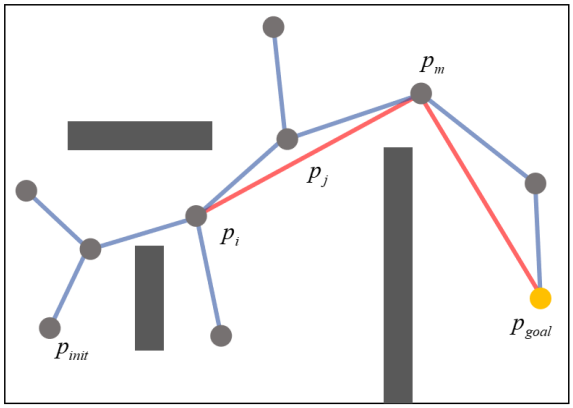


Fig. 4. Schematic diagram of node reconnection in the RRT

by the number 1, and the grid not occupied by the obstacle is represented by 0. According to the map information, the obstacle density coefficient ρ and the obstacle attenuation coefficient λ are constructed as shown in the following formula (5):

$$\rho_i = \frac{1}{|\Omega_i|}\int_{\Omega_i} I_{\text{obs}}(x,y)\,\mathrm{d}x\,\mathrm{d}y, \qquad \lambda(\rho_i) = e^{-\beta\rho_i}],. \tag{5}$$

In the formula, I_{obs} is a binary function, where the obstacle area is 1, and the free area is 0, $|\Omega_i|$ represents the area of the region, and β is the density factor. Using the obstacle density coefficient as an adjustment factor, the dynamic step-size formula is constructed as shown in formula (6) below:

$$\varepsilon(t) = \lambda_i\left[\varepsilon_{\text{base}}\cdot\left(\frac{1}{1+\tau\cdot N_{\text{fail}}}\right)\cdot\left(1+\zeta\cdot\frac{d_{\text{init}}-d_{\min}(t)}{d_{\text{init}}}\cdot\frac{1}{1+\xi\cdot N_{\text{fail}}}\right)\right], \tag{6}$$

where $\varepsilon_{\text{base}}$ is the basic step-size, $N_{\text{fail\_goal}}$ is the number of failed expansions of the expanding tree near the target point, which reflects the density of obstacles near the target point, and τ, ζ, χ are adjustment coefficients. In the formula, the density of obstacles can be adjusted by the number of collisions with obstacles through $1/(1+\tau\cdot N_{\text{fail}})$. When the density of local obstacles increases, the step size decreases accordingly. The dynamic step-size can reduce the number of iterations of the algorithm.

After obtaining the path through the improved RRT algorithm, a node re-connection strategy is adopted to eliminate redundant nodes. Its core operation is to iteratively check whether nonadjacent nodes in the path can be directly connected without colliding with obstacles. The determination and removal conditions of redundant nodes are shown in Fig. 4.

For any three consecutive nodes p_i, p_j, p_m $(i < j < m)$ in the path, if the line segment $\overline{p_i p_m}$ has no collision, then the node p_j is redundant and can be removed. The specific operation steps are shown in Algorithm 4:

Algorithm 1. Operation steps of node reconnection

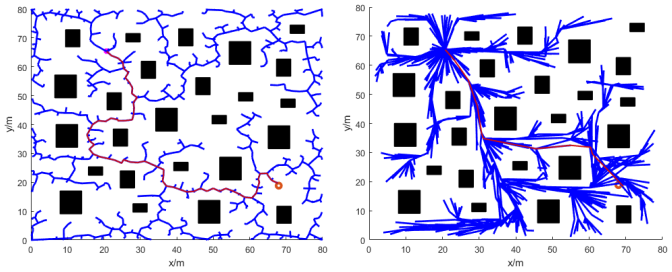
Operational steps:

1. *Initialize the reconnected path:* $Path_{rewire} = \{p_{init}\}$.
2. *Set the current node* $p_{current} = p_{init}$.
3. ***For*** *each subsequent node* p_j $(j = i+1, i+2, \ldots, k+1)$ *in the original path:*
   - *Attempt to connect* $p_{current}$ *to* p_j *with a straight line.*
   - ***If*** $\overline{p_{current}p_j}$ *is collision-free (verified by obstacle collision detection), continue to check* p_{j+1}.
   - ***Else*** *(collision occurs):*
     - *Add the previous valid node* p_{j-1} *to* $Path_{rewire}$:
       $$Path_{rewire} \leftarrow Path_{rewire} \cup \{p_{goal}\}.$$
     - *Update* $p_{current} = p_{j-1}$.
4. *Add the goal node:* $Path_{rewire} \leftarrow Path_{rewire} \cup \{p_{goal}\}$

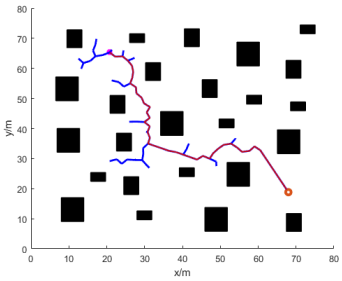
Output*: Optimized path* $Path_{rewire}$ *with minimal nodes.*

The improved algorithm with dynamic biased sampling and dynamic step-size can greatly enhance the planning speed of the algorithm and significantly reduce the redundancy of nodes. Therefore, for the optimizations of these two parts, obstacle maps are constructed to compare the RRT*, RRT, and the improved RRT algorithms. The two maps have different obstacle densities, with one map having fewer obstacles and the other having more obstacles and a more complex environment. The starting and ending positions of the three types of expanding trees are set to be the same, and the planning situations of the three algorithms are shown in the following figures. In Fig. 5 (a), (b), and (c) represent the path-planning comparisons of the RRT, RRT*, and the improved RRT algorithms in the complex-obstacle environment, respectively. The blue line segments represent the expanding trees, the red line segments represent the successfully planned paths, the red circles represent the endpoints, the purple dots represent the starting points, and the black rectangles represent the obstacles. As can be seen from the figures, in both map environments, the traditional RRT algorithm and the RRT* algorithm have numerous redundant nodes and redundant branches. Compared with the RRT algorithm, the

paths planned by the RRT* algorithm in the two maps above are superior to those of the RRT algorithm. The optimized RRT algorithm reduced a large number of redundant nodes, and the path length is shorter compared to that before optimization. Table 1 shows the comparisons of various data in the complex obstacle-map environments. Similar performance gains were observed in simple maps.



(a) Expansion situation of RRT (b) Expansion situation of RRT*



(c) Expansion situation of improved-RRT

Fig. 5. Comparison of RRT, RRT*, and Improved-RRT in the complex map

Table 1
The expansion situations of three algorithms in an environment with complex obstacles

The second group	Number of random tree nodes [N]	Number of path nodes [N]	Path length [m]	Expansion time [s]
RRT	1293	66	121.64	55.61
RRT*	697	22	86.34	20.16
Improved-RRT	86	28	82.68	1.35

As can be seen from Table 1 above, the total number of nodes, path length, number of path nodes, and expansion time of the RRT algorithm are all greater than those of the other two algorithms. The improved algorithm also shows a huge improvement in terms of time and the total number of nodes in the expanding tree. It only has slightly more path nodes than the RRT* algorithm. The comparison shows that the improved algorithm significantly reduces path-planning time and the number of redundant nodes in complex map environments, providing a solid basis for further enhancements to local path planning. A similar performance improvement is also observed in simple maps.

4. FUSION ALGORITHM

4.1. Traditional DWA algorithm

The dynamic window approach (DWA) is a widely-used local path-planning method based on predictive control theory. It predicts multiple trajectories by sampling within the vehicle's velocity space (v, ω). The feasible velocity sampling space, V_r, is constrained by the maximum/minimum velocities of the vehicle, motor performance (acceleration limits), and a safe distance from obstacles (collision limits).

Although DWA is computationally efficient, as a purely local planner, it suffers from significant drawbacks [27, 28]. First, it is highly prone to becoming trapped in local optima in complex environments (e.g., U-shaped traps), failing to find a global path. Furthermore, its ability to handle dynamic obstacles is limited, and its resulting path smoothness often requires improvement.

4.2. Optimization of the DWA algorithm

To improve the planning efficiency of DWA and enable the algorithm to achieve path planning in a dynamic environment, this paper introduces a dynamic risk assessment term based on the time-to-collision to balance the motion efficiency and collision safety. The dynamic risk term constructs an exponential penalty model of spatiotemporal correlation by traversing all dynamic obstacles in the environment, and the formula is as follows:

$$G_{\mathrm{dyn}}(v,\omega) = \sum_{i=1}^{N_{\mathrm{obs}}} k_{\mathrm{dyn}} \cdot \exp\left(-\frac{\left\|P^i_{\mathrm{obs}} - P_{\mathrm{Vehicle}}\right\|}{T_{\mathrm{collision}} \cdot \left\|v^i_{\mathrm{obs}} - v_{\mathrm{Vehicle}}\right\| + \tau}\right). \quad (7)$$

where P^i_{obs} and v^i_{obs} describe the position and velocity of the i dynamic obstacle, respectively; P_{Vehicle} and v_{Vehicle} are the current pose and velocity of the car, k_{dyn} is the weight coefficient for adjusting the dynamic risk, $T_{\mathrm{collision}}$ is the time-to-collision threshold, and τ is a very small constant to avoid a zero denominator. This evaluation function dynamically adjusts the weight value that affects the motion state of the car by establishing the relationship among the distance, velocity, and time between the car and the obstacle.

On the other hand, to address the issue that the DWA algorithm is prone to becoming trapped in local optimal solutions under complex circumstances, the global path nodes of the RRT are introduced to guide the DWA algorithm as its local target points. The specific idea is shown in Fig. 6, in which the target point and the starting point of the car are blocked by a semi-enclosed obstacle. Conventional algorithms often become locally stagnated in the semi-enclosed obstacle space in such a situation, making it impossible to plan a global path. The red path in Fig. 6 is the global path planned by the DWA algorithm with the guidance of the global nodes of the RRT. The global path of the RRT contains a total of seven local guiding nodes. Now, taking these seven local guiding nodes as the local targets of the DWA algorithm in sequence, the moving trajectory of the car can move towards the target point in the order of $p_1 \rightarrow p_2 \rightarrow p_3 \rightarrow p_4 \rightarrow p_5 \rightarrow p_6 \rightarrow p_{\mathrm{goal}}$. Through the guidance of the local target nodes, the DWA algorithm can escape the local stagnation.

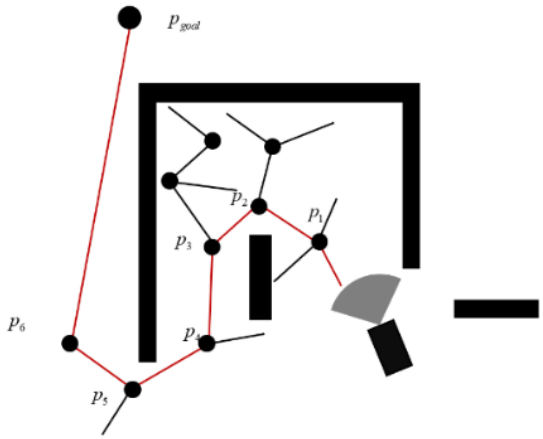


Fig. 6. The RRT nodes guide the DWA algorithm for local escape from entrapment

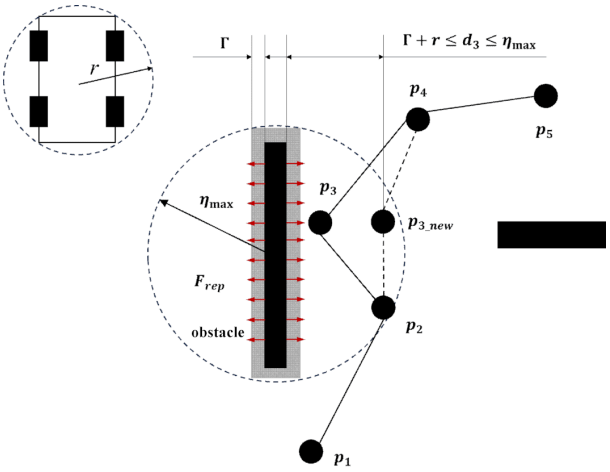


Fig. 7. Node position optimization based on repulsive potential field

During the process of guiding with RRT global path nodes, the impact of obstacles on the dynamic window algorithm needs to be considered. Since the paths planned by the random tree algorithm usually do not take into account the geometric constraints of the vehicle, the planned path points may be too close to obstacles. Although the guiding nodes of the path do not collide with obstacles, when the car moves with such a point as the local target, it may collide with obstacles due to its own geometric shape. To solve this problem, the artificial potential field method is introduced to optimize the guiding nodes that are too close to obstacles. The artificial potential field method is commonly used for obstacle-avoidance path planning of vehicles [29]. Its basic idea is to construct a repulsive potential field around obstacles and an attractive potential field around the target point. An object in the potential field is affected by the combined force of the two potential fields; that is, jointly by the repulsive and attractive forces. The resultant force of the repulsive force and the attractive force is the moving direction of the controlled object. When using the artificial potential field to increase the distance between path nodes and obstacles, since there is no need to plan a complete path, only the impact of the repulsive field and the repulsive force need to be considered. During the construction of the repulsive field, the geometric shape of the vehicle needs to be taken into account. Let us assume that the maximum geometric radius of the car is r, and the allowance of the obstacle expansion layer is Γ. The position state of the guiding node under the repulsive potential field is shown in Fig. 7.

In Fig. 7, the red arrow represents the repulsive force generated by the obstacle near the guiding node, the black dot represents the guiding node, and the dashed-line circle around the obstacle represents the influence range of the repulsive force, where $\eta_{\max}$ is the maximum radius of the repulsive field, F_{rep} is the magnitude of the repulsive force, and p_{3_new} is the rearranged position of the node p_3 under the action of the repulsive force.

The safety-distance threshold for the vehicle not to collide with the obstacle is:

$$Q^* = r + \Gamma. \tag{8}$$

When the distance d from the path node to the obstacle is $d < Q^*$, the geometry of the vehicle is likely to collide with the obstacle. In the design of the repulsive potential field, the influence of the safety distance Q^* needs to be considered. For each path node $p(x_i, y_i)$ in the global path, let us calculate its distance d_i to the nearest obstacle. If $d_i < Q^*$, then apply repulsive potential field to the node as shown in formula (9):

$$U_{\text{rep}}(p_i) = \begin{cases} \dfrac{k_{\text{rep}}}{2}\left(\dfrac{1}{d_i} - \dfrac{1}{Q^*}\right)^2 & d_i < Q^*, \\ 0, & d_i \ge Q^*. \end{cases} \tag{9}$$

In the equation, k_{rep} is the repulsive force coefficient, which is used to control the intensity of the potential field. The direction of the repulsive force is the unit vector $\vec{n} = (p_i - p_{\text{obs}})/d_{\text{obs}\_i}$ from the obstacle to the node, where p_{obs} is the position of the obstacle closest to the node. The repulsive force is the negative gradient in the repulsive potential field, representing the direction in which the repulsive potential field changes most rapidly, as shown in formula (10):

$$F_{\text{rep}}(q) = -\nabla U_{\text{rep}}(q) = k_{\text{rep}}\left(\frac{1}{d_i} - \frac{1}{Q^*}\right)\cdot\frac{1}{d_i^2}\cdot\vec{n}. \tag{10}$$

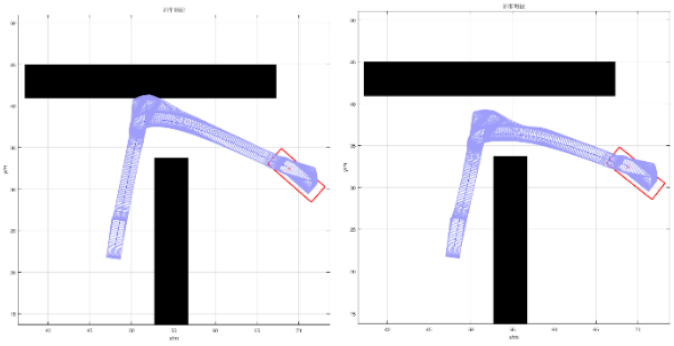
When the path node p_i moves along the direction of the repulsive force, the moving distance needs to be restricted to avoid excessive deviation from the original path. The formula for restricting the moving step-length is shown as (11):

$$p_{i\_\text{new}} = p_i + \min\left(\eta_{\max}, \left\|F_{\text{rep}}\right\|\right)\cdot\frac{F_{\text{rep}}}{\left\|F_{\text{rep}}\right\|}, \tag{11}$$

where $\eta_{\max}$ is the maximum allowable moving step-length. This formula avoids drastic movement of the node and prevents deformation of the original path.

Figure 8 shows the interference diagram of the vehicle trajectory after adding the potential field. The width of the vehicle's geometric shape is 1.8 meters, and the length is 4.72 meters. The blue trajectory represents the interference path of the vehicle's movement, and the red square represents the attitude of its end

– point parking space. As can be seen from Fig. 8a, before optimizing the guiding nodes, the geometric contour of the vehicle collided with the obstacle at the turning point. This would lead to a collision between the vehicle and the obstacle during the actual parking process, increasing the risk of injury to the occupants in the vehicle. However, in Fig. 8b, the vehicle can pass smoothly at the previous turning position. For the parking path, this means that the vehicle can pass through relatively narrow places and is less likely to cause accidents.



(a) Before APF optimization (b) After APF optimization

Fig. 8. Path comparison before and after optimization by the artificial potential field

In the global path-planning framework that integrates DWA with RRT, addressing local stagnation is essential. In addition, several aspects of the DWA algorithm require further optimization to improve its adaptability in diverse environments. First, optimization of the smoothness of the trajectory is required. As discussed above, the advantages and disadvantages of the path planned by the DWA algorithm are affected by the evaluation function. The original cost function only includes the velocity evaluation function, the obstacle evaluation function, and the evaluation function of the heading angle. In order to make the planned path smoother and thus reduce the jitter of the vehicle during tracking control, a smoothness cost function is introduced. The smoothness cost function ensures the smoothness of the path by controlling the curvature of the planned path. The smoothness cost function is as follows:

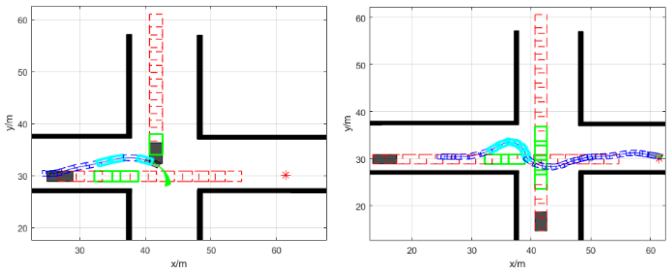
$$G_{\text{smooth}} = \frac{1}{\sum_{i=1}^{N} \left| \frac{\omega_t}{v_t} - \frac{\omega_{t-1}}{v_{t-1}} \right|}, \tag{12}$$

where v_t and ω_t represent the predicted linear velocity and angular velocity of the vehicle at time step t within the DWA trajectory generation process. This formula reflects the severity of the curvature change by calculating the difference in the rate of change of curvature between each pair of adjacent points. According to the curvature formula $\kappa = \omega/v$, the smaller the curvature change, the smoother the trajectory, and the higher the score.

Secondly, in actual planning, to improve the performance of the algorithm, the moving speed of the vehicle should be adjusted dynamically. The speed weight should be appropriately increased in open areas, and the safety weight should be increased in areas with dense obstacles. Then, when the vehicle is approaching the target point, the direction weight needs to be increased appropriately to accelerate its arrival at the target point. Finally, based on formulas (12) and the curvature formula, the weight function of the final dynamic obstacle-avoidance DWA algorithm is as follows:

$$\begin{aligned} G(v,\omega) = {} & \alpha \cdot \text{vel}J(v,\omega) + \beta \cdot \text{obs}J(v,\omega) + \ldots \\ & + \gamma \cdot \text{heading}J(v,\omega) + \delta \cdot \text{smooth}J(v,\omega) \\ & + \eta \cdot G_{\text{dyn}}, \end{aligned} \tag{13}$$

To verify the obstacle-avoidance effect of this algorithm against dynamic obstacles and its path-finding performance in complex circuitous environments, two types of obstacle environments were established, and the DWA algorithm was compared before and after optimization. The first one is a dynamic obstacle environment, as shown in Fig. 9, which is used to test the dynamic obstacle-avoidance ability of the algorithm. The second environment is a complex circuitous road, as shown in Fig. 10, which is used to test whether the algorithm experiences local stagnation.

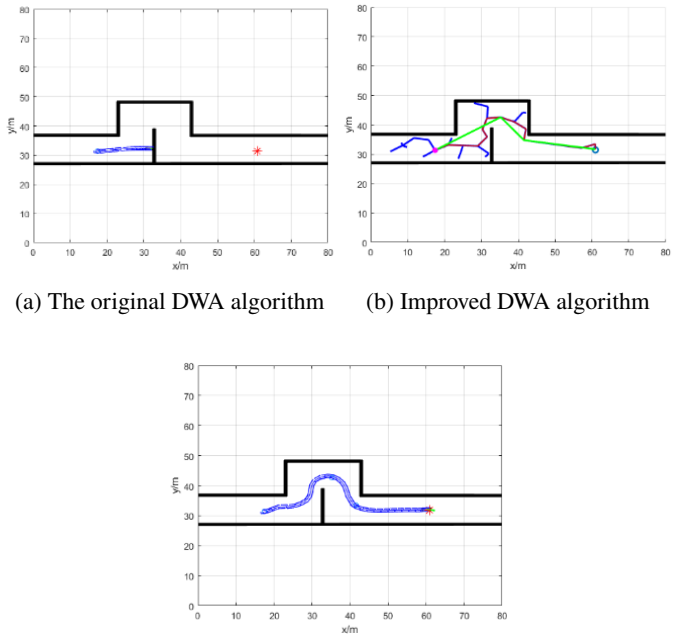


(a) The original DWA algorithm (b) Improved DWA algorithm

Fig. 9. Comparison of the DWA algorithm before and after improvement in a dynamic environment

Figures 9a and 9b, respectively, show the path-finding performances of the DWA algorithm before and after improvement in an environment with dynamic obstacles. There are two obstacles in Fig. 9, one moving from top to bottom and the other from left to right, both with a speed of 2.5 m/s. The red boxes represent the historical positions of the obstacle contours, the dark blue boxes represent the historical positions of the vehicle, the green boxes represent the historical positions of the high-risk collision areas of the obstacles, and the sky-blue boxes represent the movement trajectories of the vehicle in the high-risk areas. It can be seen that the DWA algorithm, before optimization, collided with the obstacle moving downward after avoiding the obstacle moving to the left, while the optimized algorithm steered in advance and successfully avoided the obstacles.

Figure 10a shows that the DWA algorithm before optimization is trapped in local stagnation and unable to plan a path, while 9b shows that the improved RRT has planned local guiding nodes, where the nodes of the green line segments are the local guiding nodes. Figure 10c shows that the DWA algorithm has successfully planned a path based on the guiding nodes.



(a) The original DWA algorithm (b) Improved DWA algorithm

(c) Optimized DWA algorithm

Fig. 10. Performance of the DWA algorithm before and after improvement in circuitous roads

Evidently, the optimized algorithm has a stronger adaptability to complex environments.

4.3. Global parking path

In the above content, the path planned using both the RRT and DWA algorithms is a global path, which generally does not meet the parking requirements. Therefore, the Reeds-Shepp curve (RS Curve) is used for local path planning in the local parking area. The RS curve can find the optimal trajectory of vehicle motion between any two points, and this trajectory is composed entirely of circular arcs and straight lines. In the RS curve path, the existence of cusps is allowed. To describe the path, subscripts are usually added to the path segments. Generally, the vehicle traveling direction includes six motion directions, namely forward straight-line motion, backward straight-line motion, forward left-turn, backward left-turn, forward right-turn, and backward right-turn. Any trajectory of the vehicle between two points can be represented by these six basic motions, which are represented by the following six symbols: S^+, S^-, L^+, L^-, R^+, R^-, where S represents straight-line motion, L represents left-turn, R represents right-turn, the plus sign "+" represents the vehicle forward motion, and the minus sign "−" represents the vehicle backward motion.

For the global parking path, the shift points can be calculated by judging the position of the end-point parking space and the obstacle situation. The global path and the local RS path are connected through the switch points to meet the vehicle pose at the end-point position. Figure 11 shows the motion trajectory of a parallel-parking vehicle.

The red trajectory represents the vehicle motion path. Point p is the switching point where the vehicle motion direction

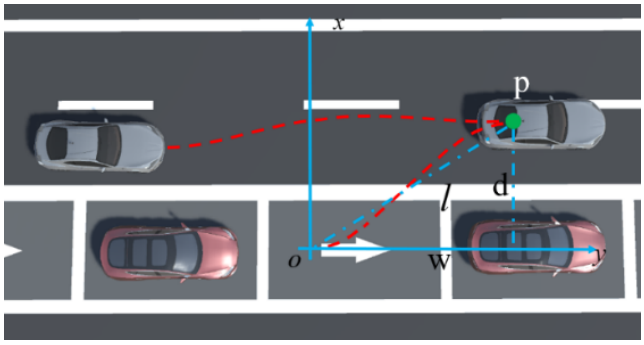


Fig. 11. The motion trajectory of a vehicle during parallel parking

changes. At this point, the vehicle can successfully reverse into the garage following the path planned by the RS curve. A local Cartesian coordinate system is established with the endpoint position. When the vehicle moves to a position parallel and aligned with the vehicle (or obstacle) on the side, the distance from the vehicle to this side-standing vehicle (or obstacle) is d, and this distance can be set according to the actual situation. In Fig. 11, it can be seen that the distance from the center point of the vehicle rear axle to the origin of the coordinate system is w, which is approximately equal to the length of the parking space. Since both d and w are known, the position of point p in the local coordinate system is (w, d). To convert this value into a point in the global coordinate system, the specific formula is shown in (14) below:

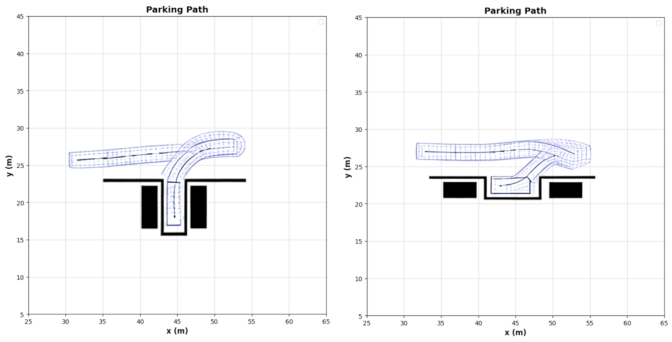
$$\begin{aligned} x_{\text{global}} &= xo + w\cos\theta - d\sin\theta, \\ y_{\text{global}} &= yo + w\sin\theta + d\cos\theta, \end{aligned} \tag{14}$$

where xo and yo is the position of the origin of the local coordinate system in the geodetic coordinate system, and θ is the angle between the x-axis of the local coordinate system and the x-axis of the global coordinate system.

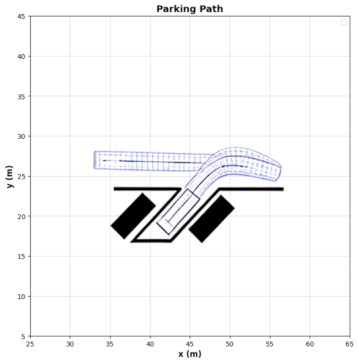
According to the above calculation method, the motion switching point p of the vehicle in the three parking scenarios can be calculated. In MATLAB, the interference trajectories of the vehicle in the three parking methods can be simulated. A rectangular box is used to simulate the vehicle contour. The width of the rectangle is 1.8 m, and the length is 4.7 m. The dimensions of the parking spaces all adopt standard sizes. The length and width of the parallel parking space and the perpendicular parking space are 6 m and 2.5 m, respectively, and the length and width of the inclined parking space are 6 m and 2.8 m, respectively. The obtained interference trajectory is shown in Fig. 12.

In Fig. 12, (a), (b), and (c) represent the perpendicular parking, parallel parking, and inclined parking scenarios, respectively. The blue box in each figure represents the outer contour of the vehicle, the black lines are the parking space lines, and the black rectangle represents the obstacle vehicle. It can be seen from the figures that in the three parking scenarios, the vehicle does not interfere with or collide with the obstacle vehicle.

To ensure effective coordination between the global reference and local execution, a target switching strategy is established.



(a) Interference path for perpendicular parking (b) Interference path for parallel parking



(c) Interference path for inclined parking

Fig. 12. Parking path interference diagram

Specifically, the Improved RRT generates a static global reference path $\mathscr{P}$, which serves as a geometric skeleton, while the improved DWA manages real-time dynamic obstacles. To prevent the vehicle from backtracking to a skipped node after avoidance maneuvers, a forward-progress check is implemented: once the vehicle projection on the global path crosses the normal line of the current target node $\boldsymbol{p}_i$, the system implicitly marks it as passed and immediately updates the local target to the next node $\boldsymbol{p}_{i+1}$, regardless of whether $\boldsymbol{p}_i$ is strictly reached. Furthermore, the heading cost function in DWA imposes a high penalty on trajectories deviating significantly from the goal direction, mathematically suppressing any reverse motion. While following the global reference path, the DWA is permitted to deviate locally to avoid moving obstacles based on the real-time dynamic risk assessment function described by (7), ensuring safety without invalidating the global plan.

5. SIMULATION VERIFICATION

To verify the reliability and authenticity of the planning algorithm, Carsim/Simulink was used to conduct a tracking control simulation of the parking path. In the simulation experiments, the three scenarios of parallel parking, perpendicular parking, and inclined parking were verified, and the vehicle tracking path, lateral error, and speed change were analyzed and discussed.

First, in this paper, a vehicle control simulation model was built through MATLAB/Simulink. For the vehicle parking control in a low-speed state, the pure pursuit control algorithm was adopted in this paper. Its core idea is to select a look-ahead point on the path and control the front wheels to steer towards the look-ahead point for driving. The formula for the front wheel steering angle is shown in formula (15):

$$\delta_f(t) = \arctan\left(\frac{2L\sin\alpha(t)}{Tv_r + l_0}\right), \tag{15}$$

where L is the wheelbase of the vehicle body, $\alpha(t)$ represents the angle between the look-ahead point and the vehicle body, which changes with time. T is the look-ahead time, v_r is the linear velocity of the center of the vehicle rear axle, and l_0 is given according to the actual working conditions.

The simulation model mainly consists of two modules, namely the path-planning module and the control module. These two modules process the vehicle's position information, and its structure is shown in Fig. 13.

To verify the feasibility of the parking paths planned by the optimized algorithm in a dynamic environment, three parking scenarios were designed. They are the parallel parking path planned on a circuitous road, the perpendicular parking path with a moving obstacle ahead, and the inclined parking path with a moving obstacle on the side. The specific parameters used in the simulation experiments are shown in Table 2.

Table 2
Specific parameters of the parking simulation experiment

Parameter category	Parameter name	Symbol	Value
Vehicle geometric parameters	Vehicle width	W_b	1.8 m
	Front overhang	l_f	0.92 m
	Wheelbase	L	2.7 m
	Rear overhang	l_r	1.1 m
	Max steering angle	$\delta_{\max}$	0.61 rad
Control parameters	Preview time	T	1.5 s
	Forward distance	l_0	1.2 s
Vehicle initial pose	Parallel	(x, y, ψ)	(11, 28, 0)
	Vertical		(23, 36, 90)
	Tilting		(11, 32, 45)
Final pose of parking space	Parallel	(x, y, ψ)	(54, 23, 0)
	Vertical		(54, 26, 0)
	Tilting		(52, 27, 0)
Dimensions of parking space	Parallel	/	(6 m, 2.5 m)
	Vertical		(2.5 m, 6 m)
	Tilting		(2.8 m, 6 m, 0.79 rad)
Lane width	Parallel	/	6 m
	Vertical		8 m
	Tilting		8 m
Dynamic obstacles	Obstacle speed	v_{obs}	2.5 m/s

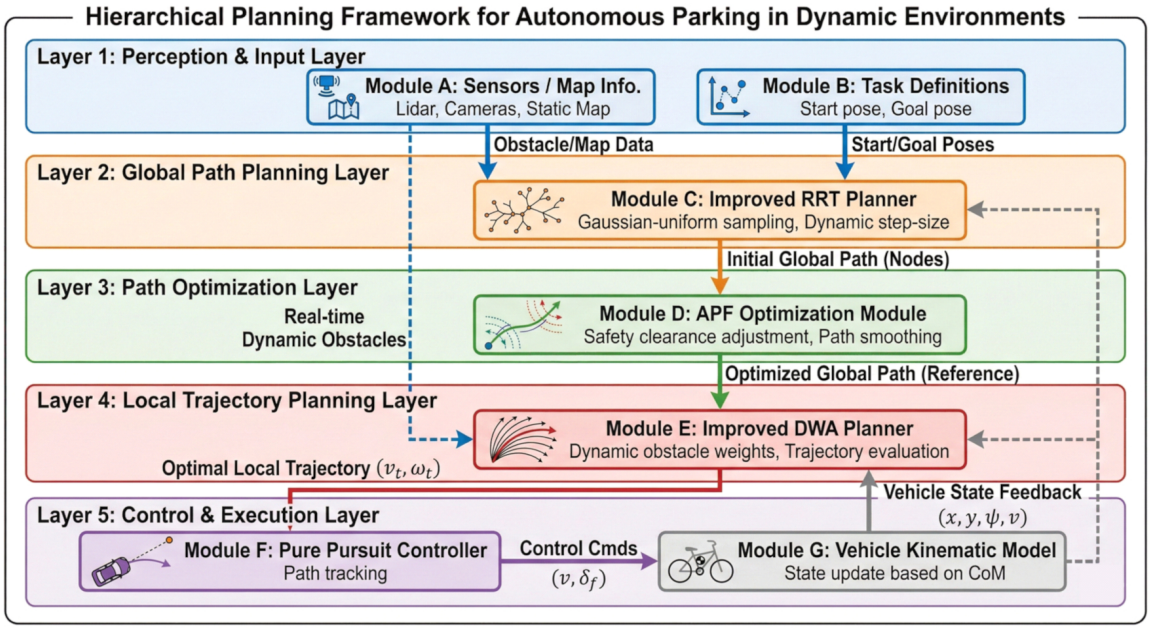


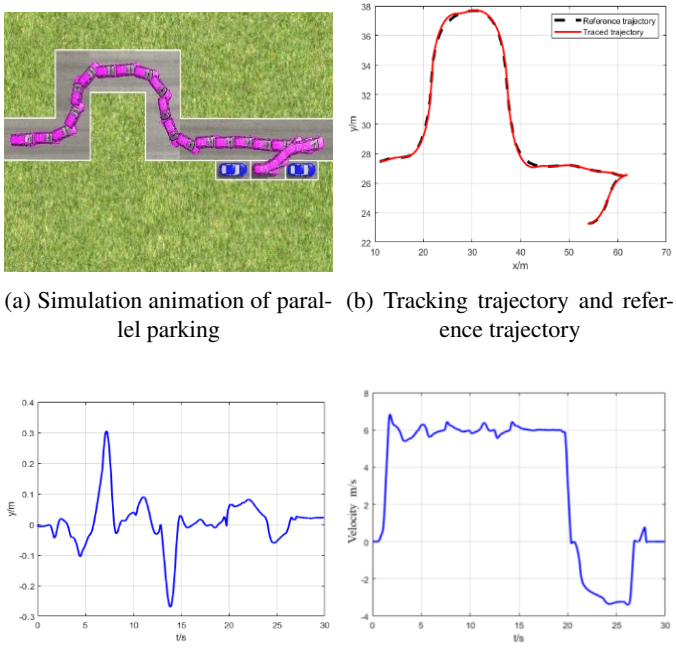
Fig. 13. Structure diagram of parking simulation

To facilitate reproducibility and clarify the configuration of the proposed framework, the key parameters used in the hierarchical planner are summarized in Table 3. These parameters were selected through empirical tuning, considering the vehicle kinematic constraints and the scale of the parking environment. For instance, the RRT step size is set to ensure that the expansion is efficient without overshooting narrow passages, while the DWA weights are balanced to prioritize collision safety in the dynamic phase.

Table 3
Key parameter settings of the proposed hierarchical path-planning framework

Parameter symbol	Description	Value	Selection basis
Global planner (RRT)			
λ	Step size for tree expansion	2.0 m	Approx. 1/2 car length
p_{goal}	Goal bias probability	0.1	Empirical (Trade-off exploration/exploitation)
Optimization (APF)			
η	Repulsive force gain	15.0	Sufficient to push nodes out of the collision zone
p_o	Influence the range of obstacles	3.0 m	Vehicle diagonal length + safety margin
Local planner (DWA)			
α	Weight for heading (azimuth)	0.05	Tuned for path tracking accuracy
β	Weight for obstacle distance	0.2	Prioritize safety in dynamic scenes
γ	Weight for velocity	0.1	Ensure smooth motion
τ	Weight for dynamic risk (TTC)	0.5	High priority for dynamic avoidance
dt	Simulation step time	0.1 s	Balance between accuracy and computation

Figure 14 below shows the simulation results of parallel parking after the vehicle passes through a circuitous bend. In the figure, (a), (b), and (c) depict the simulation animation inter-



(a) Simulation animation of parallel parking

(b) Tracking trajectory and reference trajectory

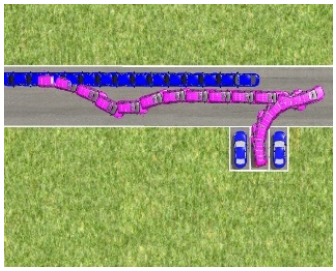
(c) Lateral error curve

(d) Speed variation curve

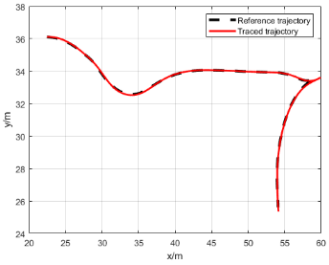
Fig. 14. Parallel parking trajectory tracking

ference diagram of the vehicle, the tracking trajectory, and the lateral error during the tracking process, respectively. The maximum lateral error in Fig. 14c is 0.31 m. Figure 14d shows the speed curve of the vehicle, with a maximum speed of 6.82 m/s. It can be seen that the vehicle can successfully park along the planned path, and both the error and the speed are within a reasonable range.

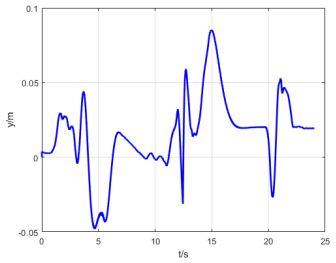
Figure 15 shows the results of perpendicular parking of the vehicle in an environment with dynamic obstacles. In Fig. 15a, the moving obstacle car moves uniformly from right to left with a speed of 2.5 m/s, as shown in Table 3. The vehicle changes its driving direction to avoid the obstacle car before a collision. The distance-change curve between the vehicle itself and the dynamic-obstacle car is shown in Fig. 15d, with a minimum distance of 4.32 m. The maximum lateral error of the vehicle is 0.072 m.



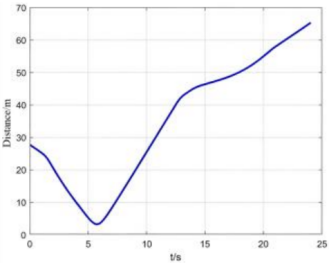
(a) Simulation animation of vertical parking



(b) Tracking trajectory and reference trajectory



(c) Lateral error curve

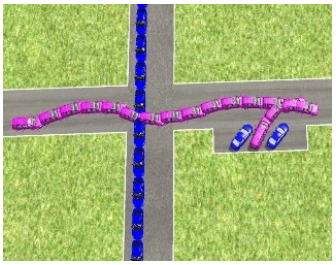


(d) The distance between the vehicle and the dynamic obstacle

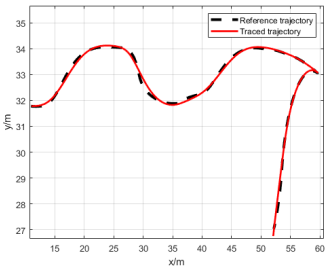
Fig. 15. Path tracking for perpendicular parking

In Fig. 16a, the obstacle car passes by the side of the vehicle. As can be seen from Fig. 16b, in order to avoid the lateral obstacle, the vehicle trajectory bends sharply, and the lateral error of the vehicle increases, reaching a maximum value of 0.73 m. The minimum distance between the vehicle and the obstacle is 2.33 m. However, considering the entire trajectory, the vehicle successfully parks in the inclined parking space without colliding with the obstacle.

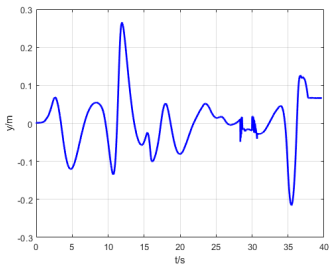
It should be noted that the "Reference trajectory", depicted as a black dashed line in Figs. 14b, 15b, and 16b represents the smoothed global path derived from the RRT nodes. By comparing the actual tracked trajectory in (a) with the reference in (b), the local deviations executed by the DWA planner for dynamic obstacle avoidance can be clearly observed.



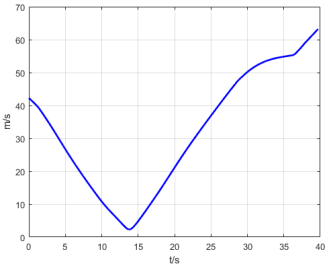
(a) Tracking trajectory and reference trajectory



(b) Tracking trajectory and reference trajectory



(c) Lateral error curve



(d) The distance between the vehicle and the dynamic obstacle

Fig. 16. Path tracking for inclined parking

5.1. Performance comparison

To demonstrate the superiority of the proposed hierarchical RRT-DWA framework, we performed a comparative analysis against state-of-the-art (SOTA) baseline methods, including the standard RRT, standard DWA, and hybrid A* algorithms. The comparison focuses on three key metrics: global search efficiency (handling local minima), path smoothness, and dynamic obstacle avoidance capability.

As shown in Table 4, the standard RRT ensures global reachability but produces rough, jerky paths unsuitable for direct vehicle tracking. Standard DWA provides smooth local control but easily falls into local minima (traps) in U-shaped obstacle environments. Hybrid A* is a powerful SOTA planner that

Table 4

Qualitative comparison of the proposed method with state-of-the-art path planning algorithms

Algorithm	Global search ability	Smoothness	Dynamic avoidance	Computational cost	Risk of local minima
Standard RRT	High	Low (Jerky)	None	Low	Low
Standard DWA	Low	High	Low (Reactive)	Low	**High**
Hybrid A*	High	High	Low (Static map)	**High**	Low
Proposed method	**High**	**High**	**High (Proactive)**	Medium	**Low**

generates kinematically feasible paths; however, it typically requires high computational resources and is originally designed for static environments, lacking real-time responsiveness to sudden dynamic threats.

In contrast, our proposed hierarchical RRT-DWA combines the benefits of global guidance (avoiding local minima) and reactive local planning (smooth control). Crucially, with the introduction of the TTC-based risk function, our method successfully handles dynamic obstacles where traditional global planners would fail. The simulation results confirm that our method achieves a 100% success rate in the tested dynamic parking scenarios with a tracking error maintained within 0.35 m.

6. CONCLUSIONS AND FUTURE WORK

To address the limitations of existing planners in dynamic environments, this study proposes a hierarchical framework integrating an improved RRT, APF, and DWA with Reeds-Shepp curves. In contrast to search-based methods like hybrid A, which often incur high computational overheads, the proposed approach balances global topological guidance with local real-time reactivity. The main contributions are summarized as follows:

Global path generation: By introducing a dynamic biased sampling strategy combining Gaussian and uniform distributions, an obstacle attenuation factor, and incorporating a dynamic step-size mechanism, the issues of a large number of redundant nodes and slow convergence in the traditional RRT algorithm are effectively resolved.

Local planning and obstacle avoidance: A collaborative RRT-DWA framework is constructed, where the RRT path (optimized by APF) provides global guidance to prevent DWA from falling into local minima. A dynamic obstacle-avoidance weight function, based on time-to-collision, is added to the DWA to solve the path-planning problem in complex dynamic environments.

Terminal maneuver: The Reeds-Shepp curve is used to smoothly adjust the final pose of the vehicle, connecting the global path to the precise parking target.

Simulation experiments demonstrate that this algorithm can successfully lead to collision-free parking in three typical dynamic scenarios: parallel, perpendicular, and diagonal parking. However, this paper still relies on simulation verification and lacks test-drive data. In the future, real-vehicle experiments will be conducted to verify the robustness and real-world applicability of the algorithm.

ACKNOWLEDGEMENTS

This work was supported by the National Natural Science Foundation of China (NSFC) [grant numbers 52502507] and by the General Scientific Research Funding Project of School of Artificial Intelligence, Zhejiang Business Technology Institute.

REFERENCES

[1] L. Cai, H. Guan, Z.Y. Zhou, F.L. Xu, X. Jia, and J. Zhan, "Parking Planning Under Limited Parking Corridor Space," *IEEE Trans. Intell. Transp. Syst.*, vol. 24, no. 2, pp. 1962–1981, Feb. 2023, doi: 10.1109/TITS.2022.3219651.

[2] K. Li, J.-G. Lu, and Q.-H. Zhang, "A Novel Scenario-based Path Planning Method for Narrow Parking Space," 2023 35th *Chinese Control and Decision Conference* (CCDC), Yichang, China, 2023, pp. 754–759, doi: 10.1109/CCDC58219.2023.10327190.

[3] S. Li and J. Wang, "Parallel Parking Path Planning in Narrow Space Based on a Three-Stage Curve Interpolation Method," *IEEE Access*, vol. 11, pp. 93841–93851, 2023, doi: 10.1109/ACCESS.2023.3310256.

[4] Y.W. Jeong, H. Wook Heo, and C.C. Chung, "A Local Path Planning for Perpendicular Parking in Limited Parking Spaces," in *Proc.* 20th *International Conference on Control, Automation and Systems* (ICCAS), Busan, Korea (South), 2020, pp. 38–40, doi: 10.23919/ICCAS50221.2020.9268290.

[5] L. Yu *et al.*, "Parallel Parking Path Planning and Tracking Control Based on Simulated Annealing Algorithm," *Int. J. Automot. Technol.*, vol. 25, pp. 867–880, 2024, doi: 10.1007/s12239-024-00087-7.

[6] D. Dong *et al.*, "Path Planning Based on A-Star and Dynamic Window Approach Algorithm for Wild Environment," *J. Shanghai Jiaotong Univ. (Sci.)*, vol. 29, pp. 725–736, 2024, doi: 10.1007/s12204-024-2731-2.

[7] T. Meng *et al.*, "Improved Hybrid A-Star Algorithm for Path Planning in Autonomous Parking System Based on Multi-stage Dynamic Optimization," *Int. J. Automot. Technol.*, vol. 24, no. 2, pp. 459–468, 2023, doi: 10.1007/s12239-023-0038-1.

[8] M. Luo, X. Hou, and J. Yang, "Surface Optimal Path Planning Using an Extended Dijkstra Algorithm," *IEEE Access*, vol. 8, pp. 147827–147838, 2020, doi: 10.1109/ACCESS.2020.3015976.

[9] N.L. Prasad and B. Ramkumar, "3-D Deployment and Trajectory Planning for Relay Based UAV Assisted Cooperative Communication for Emergency Scenarios Using Dijkstra's Algorithm," *IEEE Trans. Veh. Technol.*, vol. 72, no. 4, pp. 5049–5063, Apr. 2023, doi: 10.1109/TVT.2022.3224304.

[10] H. Zhang, Y. Wang, J. Zheng, and J. Yu, "Path Planning of Industrial Robot Based on Improved RRT Algorithm in Complex Environments," *IEEE Access*, vol. 6, pp. 53296–53306, 2018, doi: 10.1109/ACCESS.2018.2871222.

[11] P. Ren, S. Chen, and H. Fu, "Intelligent Path Planning and Obstacle Avoidance Algorithms for Autonomous Vehicles Based on Enhanced RRT Algorithm," in *Proc. 21 6th International Conference on Communication and Electronics Systems (ICCES)*, Coimbatre, India, 2021, pp. 1868–1871, doi: 10.1109/ICCES51350.2021.9489113.

[12] W. Wang, J. Li, Z. Bai, Z. Wei, and J. Peng, "Toward Optimization of AGV Path Planning: An RRT*-ACO Algorithm," *IEEE Access*, vol. 12, pp. 18387–18399, 2024, doi: 10.1109/ACCESS.2024.3359748.

[13] S. Ganesan, S.K. Natarajan, and A. Thondiyath, "G-RRT*: Goal-oriented Sampling-based RRT* Path Planning Algorithm for Mobile Robot Navigation," in *Proc. 2021 5th International Conference on Advances in Robotics (AIR'21)*, 2021, New York, USA, p. 23, doi: 10.1145/3478586.3478588.

[14] Q. Liu *et al.*, "An Improved Genetic Algorithm with Modified Critical Path-based Searching for Integrated Process Planning and Scheduling Problem," *J. Manuf. Syst.*, vol. 70, pp. 127–136, 2023, doi: 10.1016/j.jmsy.2023.07.004.

[15] K. Zheng and S. Liu, "RRT based Path Planning for Autonomous Parking of Vehicle," in *2018 IEEE 7th Data Driven Control and Learning Systems Conference (DDCLS)*, Enshi, China, 2018, pp. 627–632, doi: 10.1109/DDCLS.2018.8515940.

[16] Q. Cheng and Y. Wang, "Research on Path Planning for Automatic Parking Based on RRT* Algorithm," in *2023 International Conference on Advances in Electrical Engineering and Computer Applications (AEECA)*, Dalian, China, 2023, pp. 507–512, doi: 10.1109/AEECA59734.2023.00096.

[17] A.C. Manav and I. Lazoglu, "A Novel Cascade Path Planning Algorithm for Autonomous Truck-Trailer Parking," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 7, pp. 6821–6835, Jul. 2022, doi: 10.1109/TITS.2021.3062701.

[18] M. Kim, J. Ahn, and J. Park, "TargetTree-RRT*: Continuous-Curvature Path Planning Algorithm for Autonomous Parking in Complex Environments," *IEEE Trans. Autom. Sci. Eng.*, vol. 21, no. 1, pp. 606–617, Jan. 2024, doi: 10.1109/TASE.2022.3225821.

[19] R. Han *et al.*, "Reinforcement Learned Distributed Multi-robot Navigation with Reciprocal Velocity Obstacle Shaped Rewards," *IEEE Robot. Autom. Lett.*, vol. 7, no. 3, pp. 5896–5903, 2022, doi: 10.1109/LRA.2022.3161699.

[20] W. Zhang, N. Wang, and W. Wu, "A Hybrid Path Planning Algorithm Considering AUV Dynamic Constraints," *Ocean Eng.*, vol. 285, p. 115333, 2023, doi: 10.1016/j.oceaneng.2023.115333.

[21] X. Gong *et al.*, "A Local Path Planning Algorithm for Robots Based on Improved DWA," *Electronics*, vol. 13, no. 15, p. 2965, 2024, doi: 10.3390/electronics13152965.

[22] Y. Hu, H. Long, and M. Chen, "The Analysis of Pedestrian Flow in the Smart City by Improved DWA with Robot Assistance," *Sci. Rep.*, vol. 14, p. 11456, 2024, doi: 10.1038/s41598-024-62134-8.

[23] Y.C. Pai and J. Patton, "Center of Mass Velocity-Position Predictions for Balance Control," *J. Biomech.*, vol. 30, no. 4, pp. 347–354, 1997, doi: 10.1016/S0021-9290(96)00165-0.

[24] W. Zhou and J. Xu, "Research on Unmanned Ship Path Planning Algorithm Based on RRT* and Artificial Potential Field Method," in *2024 5th International Seminar on Artificial Intelligence, Networking and Information Technology (AINIT)*, Nanjing, China, 2024, pp. 194–199, doi: 10.1109/AINIT61980.2024.10581571.

[25] J. Xue *et al.*, "Optimal Parking Path Planning and Parking Space Selection Based on the Entropy Power Method and Bayesian Network," *Sustainability*, vol. 15, no. 11, p. 8450, 2023, doi: 10.3390/su15118450.

[26] L. Shen *et al.*, "Robust Line Segment Mismatch Removal Using Point-pair Representation," *ISPRS J. Photogramm. Remote Sens.*, vol. 203, pp. 314–327, 2023.

[27] X. Lai *et al.*, "Enhanced DWA Algorithm for Local Path Planning of Mobile Robot," *Ind. Robot*, vol. 50, no. 1, pp. 186–194, 2023, doi: 10.1108/IR-05-2022-0130.

[28] L. He, S. Zhang, H. Zhang, and L. Yuan, "Multiple Interaction Strategies for Mobile Robots Based on Improved Dynamic Window Approach," *Ind. Robot*, vol. 51, no. 1, pp. 105–116, 2024, doi: 10.1108/IR-07-2023-0152.

[29] L. Zhai *et al.*, "Local Trajectory Planning for Obstacle Avoidance of Unmanned Tracked Vehicles," *IEEE Access*, vol. 12, pp. 19665–19681, 2024, doi: 10.1109/ACCESS.2024.3355952.