Hierarchical Path Planning and Motion Control Framework Using Adaptive Scale Based Bidirectional Search and Heuristic Learning Based Predictive Control

Du, Guodong; Zou, Yuan; Zhang, Xudong; Li, Zirui; Liu, Qi

DOI
10.1109/TVT.2025.3532643

Publication date
2025

Document Version
Final published version

Published in
IEEE Transactions on Vehicular Technology

Hierarchical Path Planning and Motion Control Framework Using Adaptive Scale Based Bidirectional Search and Heuristic Learning Based Predictive Control

Guodong Du, Yuan Zou, *Senior Member, IEEE*, Xudong Zhang, *Member, IEEE*, Zirui Li, and Qi Liu

***Abstract*—Autonomous vehicles have been used for a variety of driving tasks, in which path planning and motion control are important research parts to realize the autonomous driving. A hierarchical framework consisting of path planning and motion control of the vehicle for non-specific scenarios is proposed in this paper. Firstly, the description and the formulations of the problem are given, and the corresponding models are constructed. Then, the logical construction of proposed framework is expounded with several logical associations and algorithmic improvements. The bidirectional heuristic planning with adaptive scale search is designed and incorporated with robust weighted regression algorithm to plan the optimal global path, while the multi-step predictive control method based on heuristic reinforcement learning algorithm is proposed to improve the effect of the motion control. The results show that the proposed framework for autonomous driving achieves better performance in both path planning and motion control than several existing algorithms and methods. The adaptability of hierarchical framework is demonstrated. Furthermore, the effectiveness of the hierarchical framework in real world scenario application is also validated.**

***Index Terms*—Hierarchical framework, path planning, motion control, adaptive scale based bidirectional search, heuristic learning based predictive control.**

I. Introduction

AUTONOMOUS vehicles are promoting the development of the intelligent vehicular technology (IVT) [1], and have been widely used in a variety of driving tasks [2]. Benefitting from its ability to reduce the driving burden, several studies on autonomous vehicles have been conducted in recent years [3]. The technical problems of autonomous vehicles mainly focus on the acquisition, perception, communication, planning, control and actuation [4]. Among these key technologies, the planning part and the control part determine the level of automatic driving. High-level automatic driving has the characteristics of high autonomy. Research on fully autonomous path planning and motion control to replace the driver behavior has practical significance in the development of vehicular technology.

For autonomous vehicles operating specific driving tasks, the research of planning usually refers to the path planning based on the driving requirements and global map. The planned global path will be used as a reference for the automatic driving process. Then, the research of control generally refers to the actual motion control, which generates control strategies based on the planned global path and real-time perception information. In case of sudden obstacles, the dynamic avoidance control will also be implemented during the automatic driving process. Recently, the path planning and motion control for autonomous vehicles have been recognized as the valuable research area using multiple methods [5].

In the study of path planning, various algorithms have been applied, such as sampling-based algorithm, optimization-based algorithm, potential field algorithm and graph search algorithm. The sampling based algorithm is often applied to the path planning using the ability to find the internal connectivity. The rapidly-exploring random tree (RRT) [6] and probabilistic roadmap method (PRM) [7] are regarded as representatives to realize the global path planning.

The optimization-based algorithms have become popular in the path planning research, such as particle swarm optimization (PSO) [8], genetic algorithm (GA) [9], and recurrent spline optimization (RSO) [10]. The planning problem is transformed into the value function in these optimization-based algorithms, and the optimal path is generated. However, the optimization is affected by the model complexity, and the computational burden can be heavy. In order to achieve the fast path planning in continuous space, some researchers focus on the application of the artificial potential field (APF) algorithm. The modified APF algorithm was proposed for the path planning of autonomous vehicles, and the simulation results proved the better planning

Received 10 September 2024; accepted 16 January 2025. Date of publication 22 January 2025; date of current version 20 June 2025. This work was supported in part by the National Key Research and Development Program of China under Grant 2021YFB2500900 and in part by Young Elite Scientists Sponsorship Program by CAST. The review of this article was coordinated by Prof. Kanghyun Nam. *(Corresponding author: Yuan Zou.)*

Guodong Du is with the Institute of Dynamic System and Control, ETH Zurich, 8092 Zurich, Switzerland, and also with the National Engineering Laboratory for Electric Vehicles, School of Mechanical Engineering, Beijing Institute of Technology, Beijing 100081, China (e-mail: guodongdu_robbie@163.com).

Yuan Zou, Xudong Zhang, and Qi Liu are with the National Engineering Laboratory for Electric Vehicles, School of Mechanical Engineering, and Collaborative Innovation Center of Electric Vehicles in Beijing, Beijing Institute of Technology, Beijing 100081, China (e-mail: zouyuan@bit.edu.cn; xudong.zhang@bit.edu.cn; 3120195257@bit.edu.cn).

Zirui Li is with the Department of Transport and Planning, Delft University of Technology, 2628 Delft, The Netherlands (e-mail: 3120195255@bit.edu.cn).

Digital Object Identifier 10.1109/TVT.2025.3532643

effectiveness than previous algorithms [11]. In [12], another path planning method based on improved APF algorithm was designed. However, the planned path might fall into the local optimum in some particular conditions. Even though some variants of the APF algorithm are developed to solve this problem, this class of algorithms is still limited in extreme scenarios, such as a long and narrow alleyway.

As the representative algorithm of global path planning, the graph search algorithm has been widely studied in this field. The global optimal path from start position to target position is generated by minimizing the accumulation of the value function. As a typical graph search, the Dijkstra algorithm was applied for the surface optimal path planning, and the feasibility of this algorithm was validated [13]. Because of its global optimality, the Dijkstra algorithm is generally used as the benchmark for other path planning algorithms. However, the global traversal search of this algorithm would take a lot of computational time. Therefore, a series of variations were developed to achieve faster search speed. A-star (A$*$) algorithm realized the graph search with the heuristic conception, and assigned different weights of path nodes [14]. The path planning method based on jump-A$*$ algorithm was designed, which was more efficient than traditional algorithms [15]. In addition, another typical graph search algorithm called D-star (D$*$) was designed using the heuristic improvement with dynamic search cost [16]. In [17], the D$*$ algorithm with negative edge weights was applied, and the simulation results validated its performance.

While these graph search algorithms constantly improve the solution speed, further reducing the planning time to achieve better practical applications is still necessary. Especially in the current demand for real-time autonomous driving, the shorter path planning time means a broader real-time application prospect. Besides, the initial path solved is usually not smooth enough due to the direct connection of nodes in graph search algorithms, which may negatively affect the path tracking performance of autonomous vehicles. It is valuable to apply an efficient continuous smoothing method so that the processed path can better meet the demand of path tracking control. Above all, designing an improved graph search scheme to further improve the rapidity and smoothness of path planning is a motivation of this research.

In the study of motion control, several methods were used, such as proportion-integration-differentiation (PID) method, pure pursuit (PP) method and linear quadratic regulator (LQR) method. The PID method has wide engineering applications due to its simplicity and reliability [18], which adjusts three ratio coefficients to control longitudinal and lateral motions. Han et al. [19] proposed a PID-based motion control method for the path tracking of intelligent vehicles, and the capability of this approach was verified by the experimental simulation. However, the control strategy designed by PID method has poor universality, which hardly adapts to different driving scenarios. The PP method is another popular motion control choice [20]. In [21], the modified PP method with adjustable look-ahead distance was developed. Nevertheless, the tracking accuracy of PP method is limited by path curvature and vehicle speed, so the control performance is mediocre under high velocity or large curvature conditions. Moreover, the PP method needs to strictly follow the reference path. Once the reference path fails, such as being blocked by sudden obstacles, the PP method cannot generate dynamic strategies to avoid obstacles independently. As another representative of motion control methods, the LQR method works well in several driving scenarios, which utilizes the linear state feedback [22]. However, the linear assumption of this method also restricts its application.

In recent years, the model predictive control (MPC) based motion control of autonomous vehicles has become a research hotspot. The MPC method solves the motion control sequence iteratively by using optimization algorithms in a prediction horizon [23]. An improved kinematic MPC method was proposed for the high-speed motion control of autonomous vehicles [24]. The results demonstrated that the developed controller tracked the reference path well. In [25], a randomized MPC method for autonomous driving was designed, which contained the reference path tracking and collision avoidance. Besides, other MPC methods, such as nonlinear MPC [26], have also been applied to related control problems of autonomous vehicles. Nevertheless, MPC method has the heavy computational load and the engineering application stability in the field of motion control is still being validated.

With the development of the concept of artificial intelligence (AI), machine learning algorithms are constantly applied to the motion control optimization of autonomous vehicles. Shan et al. [27] proposed a reinforcement learning (RL) based motion control strategy, which involved driving smoothness and tracking accuracy. The results proved that this RL-based control was better than MPC-based control and LQR-based control. Another adversarial RL framework was designed for the motion control and collision avoidance in the autonomous driving [28]. The effectiveness of this framework was validated by simulation tests. Besides, other variants of RL methods have also been applied to the motion control research, such as partial RL method [29], safe RL method [30], and configurable RL method [31]. Actually, the RL architecture is suitable for solving dynamic motion control problem with its powerful self-learning ability. The general RL algorithms belong to Markov decision process (MDP), and generate single-step control action according to the current state. However, the actual global optimal solution of a control sequence is often derived in a long solution time domain. The single-step decision of the existing RL method does not consider the state of the next several steps, and the generated control strategy is only optimal for the current state, which may miss the actual optimal control strategy of the current state from the perspective of global optimization. Therefore, the set of one-step solution of RL method easily falls into the problem of sub-optimal. In addition, the above motion control research mainly focuses on the path tracking problem, but does not fully consider the situation of sudden obstacles and the coupling of path tracking control and dynamic collision avoidance control.

Facing with the above problem and referring to the advantage of MPC method, the predictive time domain concept of MPC is introduced into the traditional RL algorithm, and the control optimization based on the future prediction horizon can effectively improve the performance of the global control policy. Therefore, incorporating a forward multi-step predictive decision method

with reinforcement learning structure to replace the single-step decision of traditional reinforcement learning is meaningful, which has the potential to achieve better motion control effect of autonomous vehicles.

Actually, the planning system and control system for automatic driving are related directly [32]. In the past few years, several researchers have focused on layered frameworks that include planning system and control system together [33], [34], [35]. Admittedly, the performance of hierarchical frameworks for autonomous vehicles is constantly being improved. However, the path planning system and motion control system in existing layered frameworks are still as discussed in above literature review, which have the possibility of further improvement.

The key contributions for this research are described by:

1) The novel hierarchical framework consisting of path planning and motion control for the autonomous vehicles is constructed for non-specific scenarios, which achieves complete autonomous driving without involving the driver. The proposed hierarchical framework combines planning and control reasonably, and the logical relations of these two modules complement each other to realize the function of autonomous driving, so as to replace the traditional completely independent planning and control modules scheme.
2) The bidirectional heuristic planning with adaptive scale search is designed and incorporated with the robust weighted regression algorithm to generate the global path that takes optimality, rapidity, and smoothness into account. This planning scheme combines the bidirectional heuristic search and adaptive scale search to maximize the aggressiveness of the search and further shorten the planning time in the large scene map. Meanwhile, the innovative integration of the robust locally weighted regression algorithm also makes up for the weakness that the initial path generated by the proposed aggressive planning method is not smooth enough.
3) The multi-step predictive control method based on the heuristic reinforcement learning algorithm is proposed to further improve the overall performance in safety, accuracy, comfort and rapidity of motion control. In this control scheme, the heuristic Q learning algorithm (HQL) is applied to the training of the initial motion control agent, which effectively improves the effectiveness of the training optimization compared to the traditional Q learning method. The future prediction horizon is constructed and the multi-step predictive control (MSPC) solution logic is designed, and then applied to the initial control agent to realize the transformation from single-step decision making to multi-step predictive decision making.

The organizational structure of this paper: In Section II, the description and the relevant formulations of the problem are given, and the motion control model is constructed. The hierarchical framework consisting of path planning and motion control is proposed with several logical associations and algorithmic improvements in Section III. In Section IV, the performance of the proposed framework is verified by virtual driving environment simulation and real world scenario test. Finally,

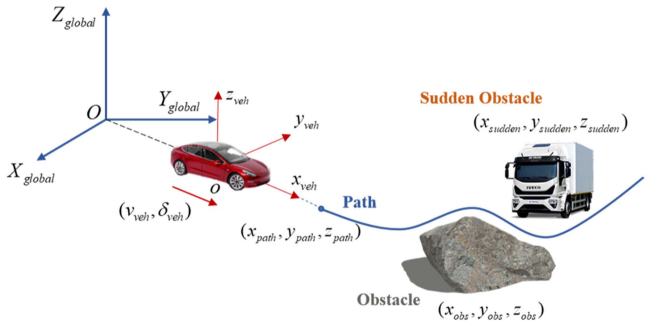


Fig. 1. The schematic diagram of autonomous driving problem.

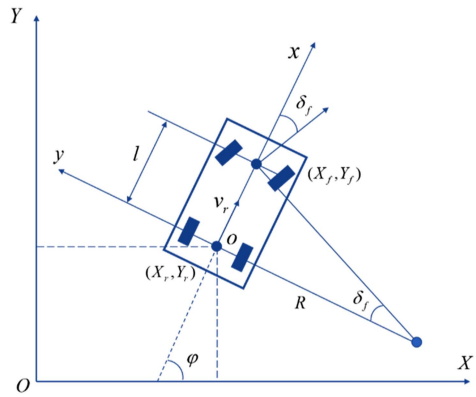


Fig. 2. The kinematic model for motion control.

Section V concludes this paper and gives the future research plan.

II. Modeling and Formulations of Path Planning and Motion Control

A. Problem Description

In the actual execution of autonomous vehicle, the globally fixed and locally dynamic information is input in rolling update. The vehicle plans the optimal global path and tracks the reference path to complete the driving task. In case of sudden obstacles, the vehicle needs to avoid collision automatically. The diagram of this problem can be illustrated in Fig. 1.

In the above schematic diagram, the inertial coordinate system and local coordinate system are represented by $O - X_{global} - Y_{global} - Z_{global}$ and $o - x_{veh} - y_{veh} - z_{veh}$, respectively. (v_{veh}, δ_{veh}) is the motion state pair of driving velocity and steering angle, which describes the driving behavior. $(x_{path}, y_{path}, z_{path})$ denotes the position of the reference path point, which is targeted for tracking control. Besides, $(x_{obs}, y_{obs}, z_{obs})$ stands for the position information of fixed obstacles, and $(x_{sudden}, y_{sudden}, z_{sudden})$ stands for the position information of sudden obstacles. In this paper, the research scene is the level road, which does not include vertical motion in the analysis.

B. Problem Modeling

Fig. 2 shows the kinematic model for motion control. *X-O-Y* and *x-o-y* stand for the inertial coordinate system and local

coordinate system, respectively. In the inertial coordinate system, the center coordinate of the front axle is represented by (X_f, Y_f), and the center coordinate of the rear axle is represented by (X_r, Y_r). The yaw angle of the vehicle is denoted by φ. In the local coordinate system, δ_f and v_r are the steering angle of front axle and velocity of rear axle center, respectively. l stands for the wheel base of vehicle, and R denotes the instantaneous turning radius of the rear axle center.

The velocity of rear axle center is calculated by:

$$v_r = \dot{X}_r \cos\varphi + \dot{Y}_r \sin\varphi \tag{1}$$

According to the constraints of axles, the kinematic equations are described by:

$$\begin{cases} \dot{X}_f \sin(\varphi + \delta_f) - \dot{Y}_f \cos(\varphi + \delta_f) = 0 \\ \dot{X}_r \sin\varphi - \dot{Y}_r \cos\varphi = 0 \end{cases} \tag{2}$$

The derivative of center position for rear axle can be related to the driving velocity as follows:

$$\begin{cases} \dot{X}_r = v_r \cos\varphi \\ \dot{Y}_r = v_r \sin\varphi \end{cases} \tag{3}$$

Besides, the front and rear axles satisfy the following geometrical equations:

$$\begin{cases} X_f = X_r + l\cos\varphi \\ Y_f = Y_r + l\sin\varphi \end{cases} \tag{4}$$

Therefore, the instantaneous turning radius and yaw velocity are derived by:

$$\begin{cases} \omega = \frac{v_r}{l}\tan\delta_f \\ \dot{\varphi} = \omega \\ R = \frac{v_r}{\omega} \end{cases} \tag{5}$$

According to the above equation, the relationship between the steering angle and the instantaneous turning radius is described as follows:

$$\delta_f = \arctan\left(\frac{l}{R}\right) \tag{6}$$

Finally, the kinematic model for the motion control can be described by the following state space equation:

$$\begin{pmatrix} \dot{X}_r \\ \dot{Y}_r \\ \dot{\varphi} \end{pmatrix} = \begin{pmatrix} \cos\varphi \\ \sin\varphi \\ 0 \end{pmatrix} v_r + \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} \omega = \begin{pmatrix} \cos\varphi \\ \sin\varphi \\ \tan\delta_f / L \end{pmatrix} v_r \tag{7}$$

The collision detection is necessary in the motion control process of the autonomous vehicle considering fixed obstacles as well as sudden obstacles. The adaptive three-circle collision detection model shown in Fig. 3 is designed in this paper.

Assuming that the driving velocity of the target vehicle is 0 m/s, the area of collision detection is the base area as shown on the right in Fig. 3. Based on the geometric center of autonomous vehicle, three circles are arranged in an array, which cover the area occupied by this vehicle. Once an obstacle enters the collision detection area, it is judged that a collision will definitely occur. Assuming that the target vehicle drives at a certain velocity, the collision detection area is dynamically adjusted

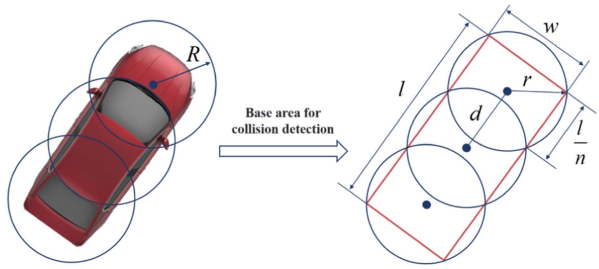


Fig. 3. The adaptive three-circle collision detection model of autonomous vehicle.

considering motion factors such as braking distance. The size of the detection area is determined by calculating the adaptive radius of circle. The adaptive radius $R_{adaptive}$ can be calculated by:

$$\begin{cases} R_{adaptive} = r + \kappa \cdot \left(v_{ini} \times t_{brk} - \frac{1}{2} \times a_{brk,\max} \times {t_{brk}}^2\right) \\ t_{brk} = \frac{v_{ini}}{a_{brk,\max}} \end{cases} \tag{8}$$

where r represents the base radius of the base area for collision detection, and κ is the weight factor of length. v_{ini} and $a_{brk,\max}$ stand for the initial driving velocity and maximum braking acceleration, respectively. The braking time t_{brk} can be obtained by v_{ini} and $a_{brk,\max}$. If no other object exists in the collision detection area, the autonomous vehicle is in a safe driving state.

C. Setting of Variables

The hierarchical framework includes the path planning layer and the motion control layer, and two sets of variables are generated. In the path planning layer, the state variables are denoted as $S_{planning} = (x_{start}, y_{start}, x_{target}, y_{target}, obc_1, obc_2, \ldots obc_{k-1}, obc_k)$, where $(x_{start}, y_{start}) \in R^2$ and $(x_{target}, y_{target}) \in R^2$ stand for the start position and target position respectively, $k \in R$ denotes the number of obstacles, $obc_k = \{(x^i_{obc}, y^i_{obc}) \mid i = 1, 2, \ldots n; (x^i_{obc}, y^i_{obc}) \in R^2\}_k$ represents the collection of the kth obstacle with plenty of obstacle pixels. Then, the continuous path planned is denoted as $P_{planning} = \{(x_{path}, f(x_{path})) \mid x_{path} \in [x_{path,\min}, x_{path,\max}]; (x_{path}, f(x_{path})) \in R^2\}$. In the path planning part, the collision detection area is adjusted to the theoretical maximum and regarded as the area occupied by the autonomous vehicle. Then, the occupied area at path position $(x_{path}, f(x_{path}))$ is given by $A_{occupied}(x_{path}, f(x_{path})) \subset R^2$. Furthermore, the passable area is represented by $A_{pass} \subset R^2$, while the impassable area is represented by $A_{impass} = R^2 \setminus A_{pass}$. Finally, the cost function of path planning is denoted as $J_{planning}(S_{planning}, P_{planning})$, and the optimal path is generated by:

$$\begin{aligned} &\arg\min J_{planning}(S_{planning}, P_{planning}) \\ &\text{s.t.} \quad \forall x_{path} \in [x_{path,\min}, x_{path,\max}] : \\ &\quad x_{path,start} = x_{start} \& x_{path,start} \in [x_{path,\min}, x_{path,\max}] \\ &\quad x_{path,target} = x_{target} \& x_{path,target} \in [x_{path,\min}, x_{path,\max}] \\ &\quad (x_{path,start}, f(x_{path,start})) = (x_{start}, y_{start}) \end{aligned}$$

$$(x_{path,target}, f(x_{path,target})) = (x_{target}, y_{target})$$
$$A_{occupied}(x_{path}, f(x_{path})) \subseteq A_{pass}$$
$$A_{occupied}(x_{path}, f(x_{path})) \cap A_{impass} = \emptyset \quad (9)$$

In the motion control layer, the path $P_{planning}$ planned by path planning layer is applied as reference for motion control of autonomous vehicle. The state variables at time i are represented by $s_i = (x_i, y_i, v_i, \Phi_i) \in S$, where $(x_i, y_i) \in R^2$ stands for the position of the vehicle, $v_i \in R$ stands for the driving velocity, and $\Phi_i \in R$ denotes the yaw angle. The sudden obstacles are applied as the disturbance for motion control:

$$\begin{cases} Obcs = obcs_1 \cup obcs_2 \cup \cdots \cup obcs_l \\ obcs_k = \{(x_{obcs}{}^i, y_{obcs}{}^i) | i = 1, 2, \ldots, \\ \quad n; (x_{obcs}{}^i, y_{obcs}{}^i) \in R^2\}_k, k = 1, 2, \cdots, l \end{cases} \quad (10)$$

where $Obcs$ represents the set of all sudden obstacles, and each sudden obstacle consists of a large number of pixels. The control variables at time i are represented by $u_i = (a_i, \delta_i) \in U$, where $a_i \in R$ denotes the acceleration, and $\delta_i \in R$ means the steering angle. Besides, the update of states is realized using discrete dynamic formulation as follows:

$$s_{i+1} = f_{\Delta t}(s_i, u_i, P_{planning}, Obcs, R_{adaptive}) \quad (11)$$

where $f_{\Delta t}$ stands for the executive function parameterized by Δt. The collision detection area $A_{det}(s_i, R_{adaptive}) \subset R^2$, the passable area $A_{pass}(s_i, Obcs) \subset R^2$, and the impassable area $A_{impass} = R^2 \setminus A_{pass}$ provide the basis to judge the safety of motion control. Finally, the optimal control strategies $u*$ is derived by minimizing the cost function $J(s_i, u_i, P_{planning}, Obcs, R_{adaptive})$:

$$u^* = \underset{s_{1:N}, u_{0:N-1}}{\arg\min} J(s_{0:N}, u_{0:N-1})$$
$$\text{s.t.} \forall i \in \{0, \ldots, N-1\}:$$
$$s_{i+1} = f_{\Delta t}(s_i, u_i, P_{planning}, Obcs, R_{adaptive})$$
$$A_{det}(s_i, R_{adaptive}) \subseteq A_{pass}(s_i, Obcs)$$
$$A_{det}(s_i, R_{adaptive}) \cap A_{impass}(s_i, Obcs) = \emptyset \quad (12)$$

III. The Hierarchical Path Planning and Motion Control Framework for Autonomous Vehicle

The planning and control of autonomous driving are closely related in terms of execution logic. The path planning system considering vehicle motion characteristics is the basis of efficient vehicle control, and the motion control system can adjust velocity and direction to avoid obstacles in real time during the driving task planned. Different from the traditional completely independent planning and control modules scheme, the collaborative operation of these two parts can ensure smooth and stable movement of the autonomous vehicle in automatic driving. The hierarchical framework proposed in this paper integrates the planning part and control part reasonably, which makes the system more adaptable to complex driving scenarios and improves the real-time performance of the system. The hierarchical framework can optimize these two layers decisions in real time to adapt to different driving scenarios. This flexibility is especially important when dealing with unknown situations and changing environment. Moreover, the hierarchical design facilitates the scalability of the framework and makes it easier to integrate path planning and motion control algorithms.

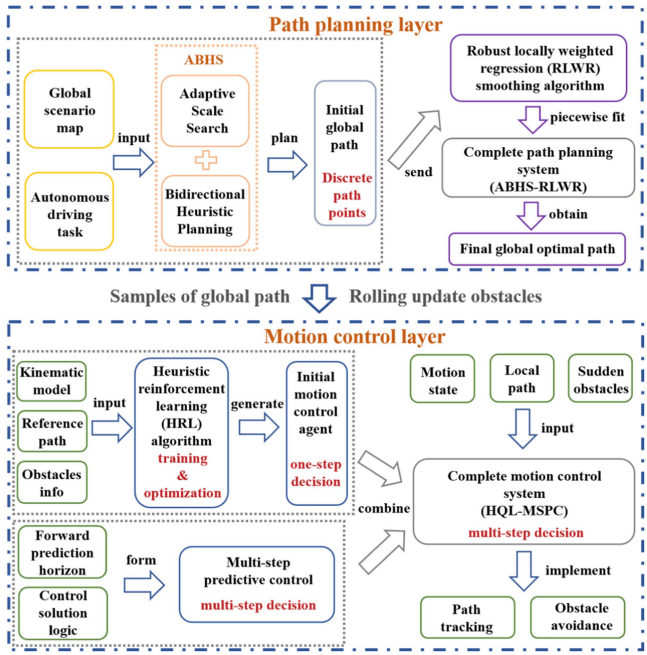


Fig. 4. The structure diagram of the hierarchical framework for autonomous vehicle.

The structure diagram of the hierarchical framework is shown in Fig. 4, which describes several logical associations and algorithmic improvements. In this hierarchical framework, the path planning layer is directly related to the motion control layer. First of all, the global scenario map is received by the path planning system and autonomous driving task is determined. Then, the adaptive scale search is proposed and combined with the bidirectional heuristic planning algorithm to plan the initial path rapidly. Afterwards, the initial discrete path is fitted piecewise by robust locally weighted regression (RLWR) smoothing algorithm, and the final global optimal path with smoothness and continuity will be obtained. Subsequently, the global path will be transmitted to the self-learning system in motion control layer for training. Based on the path reference to be tracked, the heuristic reinforcement learning (HRL) algorithm is applied to optimize the motion control agent. Then, the forward prediction conception is introduced into the reinforcement learning system, and the multi-step predictive control method based on agent optimized by HRL is proposed. Meanwhile, the local path and sudden obstacles are input into the control layer in rolling updates. Finally, the motion control strategies containing real-time obstacle avoidance will be generated.

A. Bidirectional Heuristic Planning Algorithm With Adaptive Scale Search

For large global maps, the planning efficiency of several heuristic planning algorithms such as A$*$ and D$*$ is still limited

by large number of nodes searched. In a specific path planning task, the total number of nodes searched is positively correlated with planning time. Designing a more reasonable search instead of the traditional single-step search can effectively reduce the total number of search nodes and shorten the planning time. In this paper, the aggressive degree of search is determined by the nearby environment of the current search node. For example, the search scale will be set larger if the nearby environment is an open area. Conversely, the search scale will be set smaller if some obstacles exist in the nearby environment. Therefore, the adaptive scale of search is formulated as follows:

$$S_{scale}=\begin{cases} Scale_{\max}, if(d_{near}\geq R_{\max}) \\ Scale_{\min}+\text{round}\left(\frac{d_{near}-R_{\min}}{R_{\max}-R_{\min}}\times(Scale_{\max} \\ \qquad -Scale_{\min})\right), if(R_{\max}>d_{near}>R_{\min}) \\ Scale_{\min}, if(R_{\min}\geq d_{near}>0) \end{cases} \tag{13}$$

where $Scale_{max}$ and $Scale_{min}$ represent the maximum and minimum search scales respectively. d_{near} stands for the distance from the current search node to the nearest obstacle. Besides, R_{max} and R_{min} are the maximum and minimum detection radiuses. The round(x) function is used to obtain the integer for x.

Furthermore, the traditional planning algorithms have the characteristic of one-direction search, searching from the start position to the target position. Under the condition of sufficient computational memory, the bidirectional search can further save the planning time. Therefore, the adaptive scale search proposed above will be carried out simultaneously based on the start position and the target position. When the two search paths from the start position and the target position meet in the global map, the global optimal path will be synthesized by the specific judgment rules. In this research, the heuristic planning functions of bidirectional search are expressed by:

$$\begin{cases} f_S(N_S)=g_S(N_S)+h_S(N_S) \\ f_T(N_T)=g_T(N_T)+h_T(N_T) \\ h_S(N_S)=\sqrt{(x_{N_S}-x_{target})^2}+\sqrt{(y_{N_S}-y_{target})^2} \\ h_T(N_T)=\sqrt{(x_{N_T}-x_{start})^2}+\sqrt{(y_{N_T}-y_{start})^2} \end{cases} \tag{14}$$

where $f_S(N_S)$ and $f_T(N_T)$ are heuristic functions of search from start position and from target position, respectively. $g_S(N_S)$ and $g_T(N_T)$ denote the corresponding accumulative path costs at node N_S and node N_T, respectively. $h_S(N_S)$ is the Euclidean distance from node N_S to the target position, and $h_T(N_T)$ is the Euclidean distance from node N_T to the start position. The implementation process of the bidirectional heuristic planning algorithm with adaptive scale search (ABHS) is described in Table I.

TABLE I
PSEUD-CODE OF PATH PLANNING ALGORITHM

Algorithm: ABHS

1. Initialize openlist_1, closelist_1, R_{max} and R_{min}
2. Initialize openlist_2, closelist_2, $Scale_{max}$ and $Scale_{min}$
3. Put start point in openlist_1 and target point in openlist_2
4. **if** openlist_1 & openlist_2 is not empty
5. Choose current nodes with highest priority from openlist_1 and openlist_2
6. **if** current node_1 meets current node_2 (satisfy judgement rules)
7. Trace the parent nodes from two current nodes progressively
8. Synthesize the path from start to target
9. **return** path
10. **else**
11. Pop current node_1 and current node_2 from openlist_1 and openlist_2
12. Push current node_1 and current node_2 into closelist_1 and closelist_2
13. Get the nearest distances d_1 and d_2 from current nodes to obstacles
14. **if** $d_1 > R_{max}$ ($d_2 > R_{max}$)
15. Set search step $S_{scale,1} = Scale_{max}$ ($S_{scale,2} = Scale_{max}$)
16. **elseif** $d_1 < R_{min}$ ($d_2 < R_{min}$)
17. Set search step $S_{scale,1} = Scale_{min}$ ($S_{scale,2} = Scale_{min}$)
18. **else**
19. $S_{scale,1} = Scale_{min}$ + round(((d_1 - R_{min}) / (R_{max} - R_{min}) × ($Scale_{max}$ - $Scale_{min}$)) ($S_{scale,2} = Scale_{min}$ + round(((d_2 - R_{min}) / (R_{max} - R_{min}) × ($Scale_{max}$ - $Scale_{min}$)))
20. **end**
21. **for** neighbor nodes with scales $S_{scale,1}$ and $S_{scale,2}$:
22. Cost of neighbor node: $c(ngh) = g(cur) + c(cur \text{ to } ngh)$
23. **if** neighbor node in closelist_1 (closelist_2)
24. Skip and choose next neighbor node
25. **elseif** neighbor node in openlist_1 (openlist_2) & $c(ngh) \geqslant g(ngh)$
26. Skip and choose next neighbor node
27. **elseif** neighbor node in openlist_1 (openlist_2) & $c(ngh) < g(ngh)$
28. Update $g(ngh) = c(ngh)$, $f(ngh) = g(ngh) + h(ngh)$
29. Set new parent node ngh. parent = cur
30. **else**
31. Set parent node ngh. parent = cur
32. Calculate $g(ngh) = c(ngh)$, $f(ngh) = g(ngh) + h(ngh)$
33. Push the neighbor node into openlist_1 (openlist_2)
34. **end**
35. **end**
36. **end**
37. **end**

B. Robust Locally Weighted Regression Smoothing

The proposed planning algorithm greatly reduces planning time, but the obtained path is discrete and the smoothness can be further optimized. Considering actual path tracking requirements of the autonomous vehicle, the RLWR method is developed to realize path continuity and improve path smoothness. As stated above, the discrete path will be divided into appropriate intervals for piecewise fitting which satisfies the properties of functions. The relevant fitting function is expressed by:

$$f_{fitting}(x)=\sum_{j=0}^{n}\psi_j x^j \tag{15}$$

where ψ_j means the coefficient of jth degree term of n-order polynomial. To solve all coefficients of the fitting function, the optimization objective is shown below:

$$\min_{\psi}\sum_{i=1}^{N}K_l(x,x_i)\left(y_i-\sum_{j=0}^{n}\psi_j x^j\right) \tag{16}$$

where (x_i, y_i) stands for the ith original path point. $K_l(x, x_i)$ denotes the weight function of distance, formulated by:

$$\begin{cases} K_l(x,x_0)=D(\frac{\|x-x_0\|}{l}) \\ D(t)=\begin{cases}(1-|t|^3)^3, & |t|<1 \\ 0, & |t|\geq 1\end{cases} \end{cases} \tag{17}$$

where $D(t)$ represents the cubic kernel function, and l is the metric window size for the kernel. Based on (16), its matrix form can be expressed by:

$$\min_{\Psi} (Y - X\Psi)^T K_l (Y - X\Psi) \tag{18}$$

Then, the set of coefficients of n-order polynomial is obtained by the following equation:

$$\Psi = (X^T K_l X)^{-1} X^T K_l Y \tag{19}$$

To ensure the robustness of fitting, the residual error of fitting is calculated on the basis of the first locally weighted regression:

$$e_i = y_i - f_{fitting}(x_i) \tag{20}$$

The double square kernel function $B(t)$ is used to obtain the weight of residual error:

$$\begin{cases} \delta_i = B(\frac{e_i}{\eta \cdot s}) \\ B(t) = \begin{cases} (1-t^2)^2, & |t| < 1 \\ 0, & |t| \geq 1 \end{cases} \end{cases} \tag{21}$$

where η is the outlier factor, and s denotes the median of the absolute value of residual error. Then, $K_l(x, x_i)$ is multiplied by the weight of residual error δ_i to generate the robust distance weight function as follows:

$$K_r(x, x_i) = \delta_i \times K_l(x, x_i) \tag{22}$$

Afterwards, the process shown in (18) and (19) will be repeated based on $K_r(x, x_i)$. Finally, the continuous global optimal path $P_{planning} = \{(x, f(x)) \mid y_{start} = f(x_{start}); y_{target} = f(x_{target}); (x, f(x)) \in R^2\}$ will be generated.

C. Heuristic Q-Learning Algorithm

The continuous global path planned in path planning layer is transmitted to the reinforcement learning system in motion control layer as training samples. Relying on the ability of self-learning, the heuristic Q-learning algorithm (HQL) is proposed to optimize the initial motion control agent which contains path tracking and collision avoidance. The heuristic Q-learning structure can be divided into Q-learning algorithm and heuristic experience replay.

Q-learning algorithm has the characteristics of model-free. The dynamics model of the autonomous vehicle may be affected by many complex factors, and Q-learning can learn without the explicit model. Besides, the environment of the autonomous vehicle is dynamic, complex, and difficult to model in advance. Q-learning algorithm can make the vehicle adapt to different driving scenarios by constantly interacting with the environment. Meanwhile, the decision space of this motion control problem is large, and Q-learning can effectively deal with this high-dimensional and continuous action space, and select the optimal action by learning the value function.

Actually, the motion control can be classified as a sequential decision process, and the value of implementing specific controls in specific states can be evaluated. In reinforcement learning algorithms, the evaluation value is generally represented by the expectation of future cumulative rewards, assuming that the specific control is implemented and optimal controls are followed thereafter [36]. Consequently, the optimal value function for control u at state s can be expressed by:

$$V^*(s, u) = \max_{\pi^*} E\left[\sum_{t=t_0}^{t=t_f} \gamma^t r(s_t, u_t) | s_{t_0} = s, u_{t_0} = u; \pi^*\right] \tag{23}$$

where π^* means the follow up optimal policy, and $E(x)$ represents the expectation function of x. γ denotes the discount factor which can balance the reward weights at different steps, and $r(s_t, u_t)$ represents the reward function formulated as follows:

$$\begin{aligned} r(s_t, u_t) = k_{acu} \cdot f_{tra}(s_t, u_t, P_{planning}) + k_{saf} \\ \cdot f_{obs}(s_t, u_t, Obcs, R_{adaptive}) \\ + k_{rap} \cdot f_{vel}(s_t, u_t, v_{\max}) + k_{com} \cdot f_{dri}(u_t) \end{aligned} \tag{24}$$

where k_{acu}, k_{saf}, k_{rap} and k_{com} stand for the accuracy factor, safety factor, rapidity factor and comfort factor, respectively. f_{tra}, f_{obs}, f_{vel} and f_{dri} represent the tracking error function, obstacle avoidance function, velocity evaluation function and motion comfort function, respectively.

According to the expression of the optimal value function $V^*(s, u)$, Q matrix (QM) with strong approximation ability is applied to approximates the optimal value function by constant training. The relevant function based on RL is described by:

$$Q^*(s_t) = \max_{u_t}(r(s_t, u_t) + \gamma Q^*(s_{t+1})) \tag{25}$$

where s_t and s_{t+1} denote the current and next states respectively. u_t denotes the currently executed control action. Then, the optimal control strategy $\pi^*(s_t)$ is derived by the following formula:

$$\pi^*(s_t) = \arg\max_{u_t}(r(s_t, u_t) + \gamma Q^*(s_{t+1})) \tag{26}$$

By solving the single-step optimal motion control strategy iteratively, the sequence of optimal control strategies will be derived. Besides, the target and optimal value functions of motion control are reformulated as follows:

$$\begin{cases} Q(s_t, u_t) = r(s_t, u_t) + \gamma \max_{u_{t+1}} Q(s_{t+1}, u_{t+1}; \xi_t) \\ Q^*(s_t, u_t) = \max_{u_t}(r(s_t, u_t) + \gamma \max_{u_{t+1}} Q(s_{t+1}, u_{t+1}; \xi_t)) \end{cases} \tag{27}$$

where ξ_t represents the set of weights in Q matrix. The update of Q matrix is expressed by:

$$\begin{aligned} Q(s_t, u_t; \xi_t') \leftarrow Q(s_t, u_t; \xi_t) + \alpha\big[r(s_t, u_t) \\ + \gamma \max_{u_{t+1}} Q(s_{t+1}, u_{t+1}; \xi_t) - Q(s_t, u_t; \xi_t)\big] \end{aligned} \tag{28}$$

where α and γ stand for the learning rate and discount factor, respectively. The update of Q matrix is realized by the update of weights ξ_t, and the perfection of weights indicates the perfection of Q matrix.

As another important part of reinforcement learning structure, the experience replay determines the training efficiency of reinforcement learning system. The training experiences are temporarily stored in the replay buffer, then they are sampled in batches to optimize the Q matrix. In previous research, the

original experience replay (ER) selects samples randomly and ignores differences in the importance of different samples. To address this weakness, the prioritized experience replay (PER) is proposed, which selects samples according to the priorities of different samples. However, PER still involves the sampling probability, which weakens the pertinence of experience replay to a particular training sample. Therefore, the heuristic experience replay (HER) is developed to match the Q-learning algorithm and improve the training efficiency. In the reply buffer, some of the more valuable samples need to be selected more frequently for the matrix training, which can speed up the training and optimization. Generally, the training value of the sample is positively correlated with its temporal difference (TD) error, and larger TD error indicates more obvious training effect. TD error is an important index in RL system, quantifying the disparity between the target and actual value functions for a specific experience sample. Based on the expression of TD error, the heuristic function $H(s_t, u_t)$ is formulated by:

$$\begin{cases} H(s_t,u_t)=\kappa(s_t,u_t)\cdot TD(s_t,u_t)=\kappa\cdot|Q^{Target}(s_t,u_t) \\ \qquad\qquad -Q^{Actual}(s_t,u_t)| \\ Q^{Target}(s_t,u_t)=r(s_t,u_t)+\gamma\max\limits_{u_{t+1}}Q(s_{t+1},u_{t+1};\xi_t) \\ Q^{Actual}(s_t,u_t)=Q(s_t,u_t;\xi_t) \end{cases} \tag{29}$$

where $\kappa(s_t, u_t)$ represents the training attenuation factor, which decreases as training times of the specific sample. Afterwards, the sampling process of HER is performed as follows:

$$\begin{cases} (s_{n_1},u_{n_1})=\arg\max\limits_{n_1}\{H(s_t,u_t)|t=[1,n_{buffer}]\} \\ (s_{n_2},u_{n_2})=\arg\max\limits_{n_2}\{H(s_t,u_t)|t=[1,n_{buffer}]\&t\neq n_1\} \\ (s_{n_3},u_{n_3})=\arg\max\limits_{n_3}\{H(s_t,u_t)|t=[1,n_{buffer}]\&t\neq n_1,n_2\} \\ \qquad\qquad\vdots \\ (s_{n_n},u_{n_n})=\arg\max\limits_{n_n}\big\{H(s_t,u_t)|t=[1,n_{buffer}]\&t\neq n_1, \\ \qquad\qquad n_2,\ldots,n_{n-1}\big\} \\ B_{training}=\{(s_{n_1},u_{n_1}),(s_{n_2},u_{n_2}),(s_{n_3},u_{n_3}),\cdots,(s_{n_n},u_{n_n})\} \end{cases} \tag{30}$$

where $B_{training}$ denotes the sampling batch of these selected experiences. n and n_{buffer} represent the number of samples for the matrix training and the size of replay buffer, respectively. Besides, in order to pursue the pertinence of training while taking into account the comprehensiveness of training, the prioritized experience replay is also used at intervals to assist the HER method. In the sampling process of PER, the probability of each experience sampled is described by:

$$p_t=\frac{(H(s_t,u_t)+\sigma)^\mu}{\sum_{k=1}^{k=n_{buffer}}(H(s_k,u_k)+\sigma)^\mu} \tag{31}$$

where μ stands for the priority factor, and σ is the division protection constant.

Furthermore, the generalized correlation coefficient (GCC) is introduced into HQL to evaluate the completeness of Q matrix training. The working mechanism of GCC is to evaluate the similarity between the matrix at current time node and the matrix at last time node. The training of the matrix is judged complete when the value of GCC is equal to 1. The expression of GCC ρ is formulated by:

$$\rho(Q_{cur},Q_{la})=\frac{\mathrm{tr}[\mathrm{cov}(Q_{cur},Q_{la})]}{\{\mathrm{tr}[\mathrm{cov}(Q_{cur})]\times\mathrm{tr}[\mathrm{cov}(Q_{la})]\}^{1/2}} \tag{32}$$

where Q_{cur} and Q_{la} represent the Q matrices of current and last time nodes. tr(X) represents the trace function of matrix X, and cov() is the covariance matrix as follows:

$$\begin{cases} \mathrm{cov}(Q_{cur})=\frac{[Q_{cur}-E(Q_{cur})]^T\times[Q_{cur}-E(Q_{cur})]}{n_{node}-1} \\ \mathrm{cov}(Q_{la})=\frac{[Q_{la}-E(Q_{la})]^T\times[Q_{la}-E(Q_{la})]}{n_{node}-1} \\ \mathrm{cov}(Q_{cur},Q_{la})=\frac{E\{[Q_{cur}-E(Q_{cur})]^T\times[Q_{la}-E(Q_{la})]\}}{1} \end{cases} \tag{33}$$

where $E(X)$ denotes the expectation function of matrix X, and n_{node} is the time node number.

D. Multi-Step Predictive Control Based on HQL Agent

The HQL agent can be directly used in the motion control, which generates the single step action based on the state. However, as described in Section I, the optimal solution of control sequence is often derived in a long horizon, and the single-step solution of HQL is generally suboptimal. To enhance control performance of HQL even further, the multi-step predictive control (MSPC) method is proposed and combined with HQL algorithm in this paper.

In the process of MSPC, the future horizon is built on the decision system of HQL. Specially, all possible control strategies are traversed for the first control action in the future prediction horizon. Then, the proposed HQL method iteratively generates follow-up control strategies for multiple steps in the whole prediction horizon starting from the current state. The cumulative rewards for diverse control sequences derived will be evaluated and tested, and the first action from sequence with the highest cumulative reward is determined for the current-state motion control.

The cumulative reward function of the future horizon can be expressed:

$$\begin{cases} J(s_k,u_k)_{h_p}=Q(s_k,u_k)+\sum_{i=k+1}^{k+h_p}Q^*(s_i)=Q(s_k,u_k) \\ \qquad\qquad +\sum_{i=k+1}^{k+h_p}\lambda_i\max\limits_{u_i}Q(s_i,u_i) \\ s_{i+1}=f_{\Delta t}(s_i,u_i,P_{planning},Obcs,R_{adaptive}), \\ \qquad\qquad i\in[k,k+h_p-1] \end{cases} \tag{34}$$

where h_p is the size of the prediction horizon. λ_i denotes the prediction attenuation factor negatively correlated with prediction progress, and $f_{\Delta t}$ stands for the executive function parameterized by Δt. Based on the above cumulative reward function, the optimal motion control strategy on state s_k is obtained by the following formula:

$$\begin{aligned} u^*(s_k)=&\arg\max_{u_k}J(s_k,u_k)_{h_p}=\arg\max_{u_k}\Bigg[Q(s_k,u_k) \\ &+\sum_{i=k+1}^{k+h_p}\lambda_i\max_{u_i}Q(f_{\Delta t}(s_{i-1},u_{i-1}),u_i)\Bigg] \end{aligned} \tag{35}$$

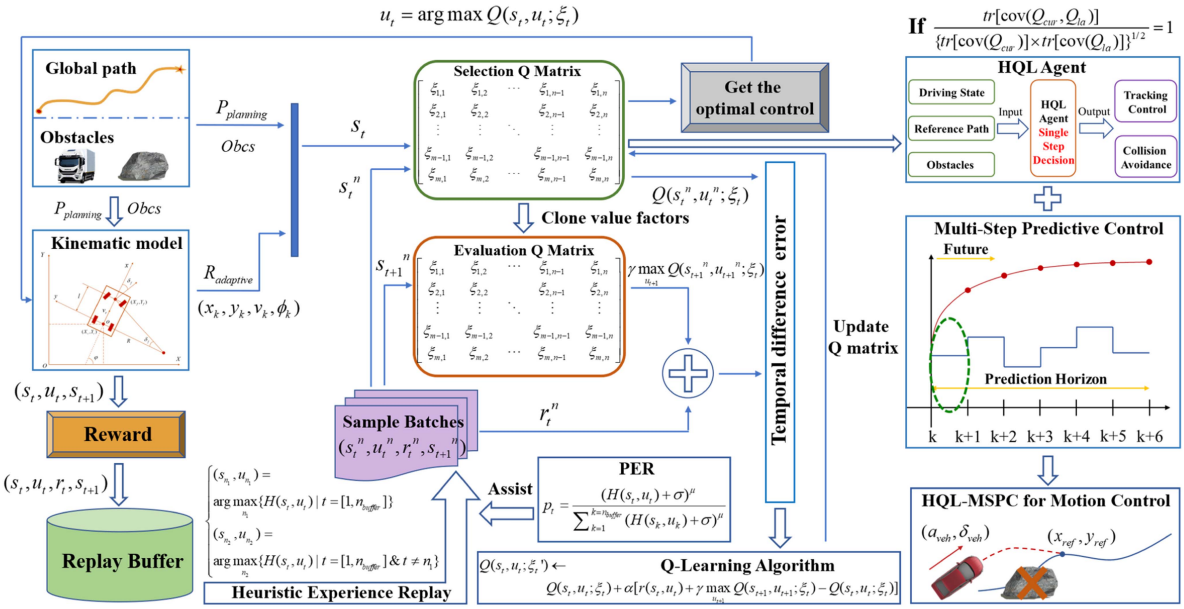


Fig. 5. The complete framework of the multi-step predictive control based on HQL agent.

TABLE II
MAIN PARAMETERS OF HQL-MSPC METHOD

Names	Values
Learning rate α	0.955
Discount factor γ	0.25
Initial training attenuation factor κ	1.0
Size of sampling batch n	23
Priority factor μ	0.75
Division protection constant σ	0.0001
Replay buffer capacity N	5000
Initial exploration factor ε	0.6
Exploration attenuation rate υ	0.98
Length of prediction horizon h_p	6
Initial prediction attenuation factor λ	0.5

The implementation process of the multiple steps predictive control is combined with well-trained HQL agent instead of other optimization algorithms and complex motion control model. Therefore, the proposed method has the sufficient solution speed and the ability of real time application. The complete framework of the multi-step predictive control based on HQL agent (HQL-MSPC) for the motion control problem is illustrated in Fig. 5. In addition, the important parameters for HQL-MSPC are listed in Table II.

IV. THE RESULTS VALIDATION AND ANALYSIS

In order to validate and evaluate the effectiveness of the proposed hierarchical framework for autonomous driving, some existing classical algorithms and methods are used for comparisons. The path planning layer and motion control layer are analyzed in subsections. In the following statement, the bidirectional heuristic planning with adaptive scale search incorporated with RLWR algorithm is called ABHS-RLWR, and the multi-step predictive control method based on heuristic Q-learning algorithm is called HQL-MSPC.

In this research, the numerical simulation, virtual driving simulation and experiment test collected from the real world scenario are carried out. The numerical simulation was completed in the workstation equipped with Matlab, and the construction of the virtual driving environment was realized based on Simulink and Driving Scenario Designer toolbox. The real vehicle collecting the real world scenario was equipped with global positioning system (GPS) and simultaneous localization and mapping (SLAM).

A. Validation of Path Planning Results

In this subsection, Dijkstra algorithm, A star (A$*$) algorithm proposed in [14] and fast A star algorithm proposed in [15] are compared with the proposed ABHS-RLWR algorithm in terms of optimality, rapidity, and smoothness. According to the received global map containing obstacles, the paths planned by these four algorithms are illustrated in Fig. 6. The constructed external environment consists of several obstacles and a long tunnel as shown in the blue ellipse. The red star and the green diamond represent the start and target positions, respectively. The area covered in black belongs to the obstacles. As Dijkstra algorithm is used for the benchmark, the trajectory of the path planned is the comparison reference. As can be seen, the path trajectory of ABHS-RLWR algorithm is similar to that of Dijkstra algorithm, which indicates that the proposed ABHS is feasible and close to optimal. The path of the proposed algorithm is smoother and more continuous than the other three paths, which

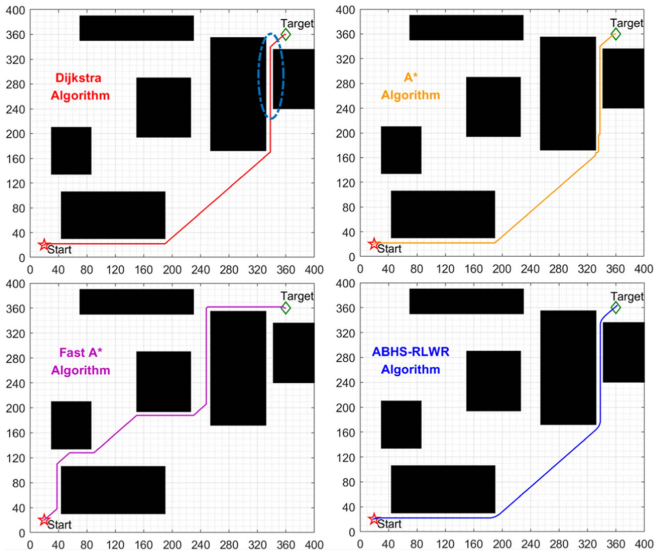


Fig. 6. The global paths planned by these four algorithms.

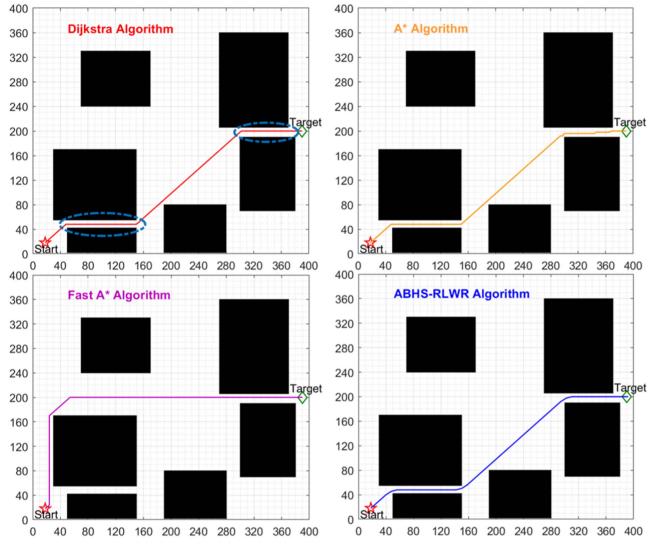


Fig. 7. The global paths planned by these four algorithms for adaptability verification.

TABLE III
KEY INDICATORS OF PATHS PLANNED BY FOUR ALGORITHMS

Algorithms	Total Length (m)	Planning Time (s)	Maximum Cumulative Curvature
Dijkstra	582.41	17.39	6.86
A star	585.59	10.73	8.29
Fast A star	616.87	6.55	8.72
ABHS-RLWR	583.50	5.36	1.60

TABLE IV
KEY INDICATORS OF PATHS PLANNED FOR ADAPTABILITY VERIFICATION

Algorithms	Total Length (m)	Planning Time (s)	Maximum Cumulative Curvature
Dijkstra	447.39	13.95	4.15
A star	449.56	8.66	6.44
Fast A star	502.91	5.02	2.72
ABHS-RLWR	447.97	3.96	1.97

proves the effectiveness of the RLWR. Besides, the path planned by the fast A∗ algorithm is completely different from the other paths. This is because the fast A∗ algorithm focuses on the speed of planning rather than the length of the path, and it takes less planning time compared with Dijkstra algorithm and traditional A∗ algorithm.

To further evaluate the algorithm, some key indicators are compared in Table III. Obviously, the length of path derived by ABHS-RLWR is shorter than those of paths planned by traditional A∗ algorithm and fast A∗ algorithm, and close to the shortest length from Dijkstra. The planning time of the proposed ABHS-RLWR algorithm is the least. Specially, the proposed algorithm takes much less calculation time than Dijkstra algorithm while ensuring the planning optimality. Although fast A∗ algorithm focuses heavily on the planning speed, the proposed ABHS-RLWR algorithm is still faster in path planning, which benefits from the bidirectional heuristic conception and adaptive scale search. Besides, the cumulative curvature is introduced to evaluate smoothness of different paths, with smaller curvature values representing higher smoothness. Apparently, the cumulative curvature of ABHS-RLWR algorithm is much smaller than those of other algorithms, further proving the effectiveness of RLWR. Therefore, the effects of the ABHS-RLWR algorithm are verified.

Second scenario is created for the adaptability verification of the proposed planning algorithm, which contains several obstacles and two tunnels. For the new global map, the paths derived by these four algorithms are illustrated in Fig. 7. The trajectory of ABHS-RLWR is still similar to that of Dijkstra algorithm. The path of the proposed algorithm is still smoother and more continuous than the other three paths. Furthermore, the relevant results of key indicators for four planning algorithms are shown in Table IV. The length of ABHS-RLWR algorithm still approaches the shortest length from Dijkstra, and shorter than those of paths planned by traditional A∗ algorithm and fast A∗ algorithm. Besides, the planning time of the proposed algorithm is the least, and the cumulative curvature of the proposed algorithm is also the smallest. Therefore, the planning result of this scenario is consistent with results of first scenario.

B. *Validation of Motion Control Results*

The motion control part contains the global path tracking and local collision avoidance. Firstly, the vehicle is controlled to track the path from the start position to the target position after receiving the global path. During the process of path tracking, the real-time obstacle avoidance strategy is generated while receiving local external environment changes in rolling update. In this subsection, the pure pursuit method with lattice planner (PP-LP) proposed in [20], the classical Q-learning method (QL) proposed in [37] and the heuristic Q-learning method (HQL) are

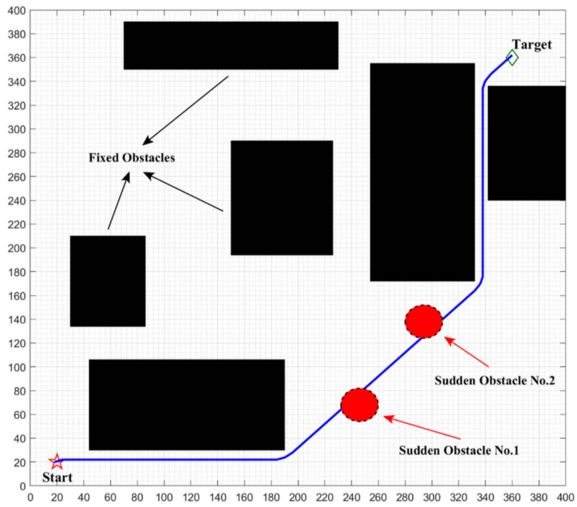


Fig. 8. The dynamic scenario with obstacles suddenly appearing.

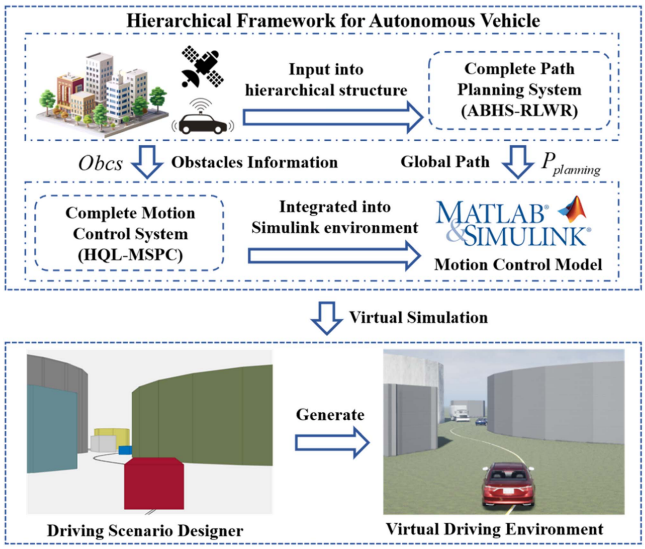


Fig. 9. The structure diagram of virtual environment simulation and platform.

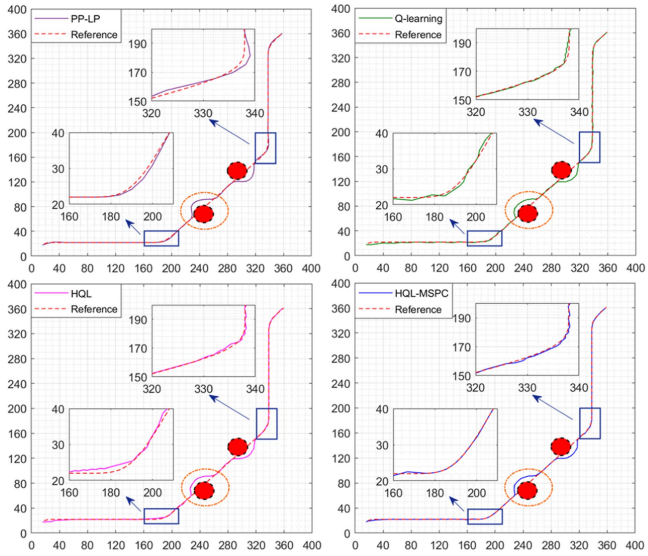


Fig. 10. The motion control results of these four control methods.

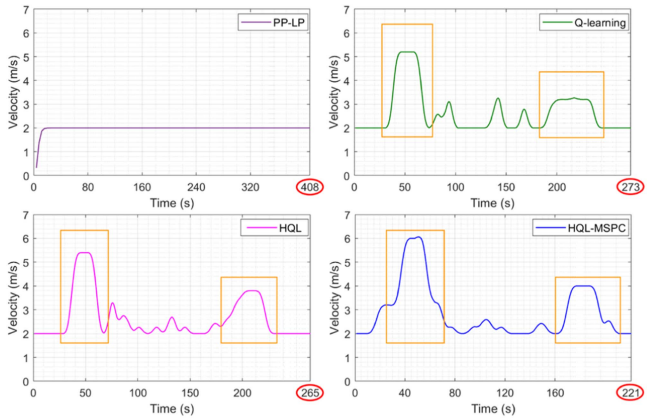


Fig. 11. The curves of driving velocities for the vehicle controlled by different methods.

compared with the proposed HQL-MSPC method in terms of safety, accuracy, comfort and rapidity. Based on the global map shown in Fig. 6, the dynamic scenario is constructed as shown in Fig. 8. The sudden obstacles block the planned path, and the autonomous vehicle needs to avoid these obstacles locally.

To evaluate the motion control performance of different methods, the virtual driving environment platform is built and the relevant simulation is carried out. The structure of virtual environment simulation and platform is shown in Fig. 9, which is based on the hierarchical framework of path generation and motion control shown in Fig. 4.

The motion control results containing path tracking and obstacles avoidance are illustrated in Fig. 10. Obviously, all four control policies complete path tracking and obstacles avoidance successfully. From the local magnification view of the blue rectangles, the tracking accuracy of the proposed HQL-MSPC method is the highest compared with other three methods in the whole motion control process. In the relatively small turning radius area, the tracking error of the PP-LP method with simple tracking rules is the largest. The better tracking performance of the proposed method than the HQL method proves the effectiveness of the multi-step predictive control (MSPC). Besides, the autonomous vehicle controlled by the HQL-MSPC method can safely avoid sudden obstacles and quickly return to the reference path. Its local collision avoidance trajectory is smooth as shown in the orange ellipse. Therefore, applying the future horizon decision in the RL solution system to replace single step decision is feasible and effective.

Fig. 11 illustrates driving velocities of the vehicle controlled by four methods. In the process of PP-LP, the target velocity is controlled at 2 m/s by PID method to ensure better motion control effect. The other three learning-based methods control the vehicle with a minimum velocity set at 2 m/s. As we can see, the autonomous vehicle drives faster in the part without sudden obstacles, while it slows down to avoid collision in the part with sudden obstacles. In particular, the driving velocity of

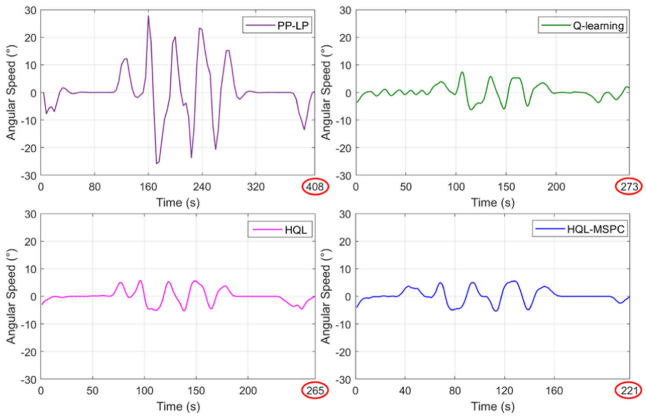


Fig. 12. The curves of steering angular speeds for the vehicle controlled by different methods.

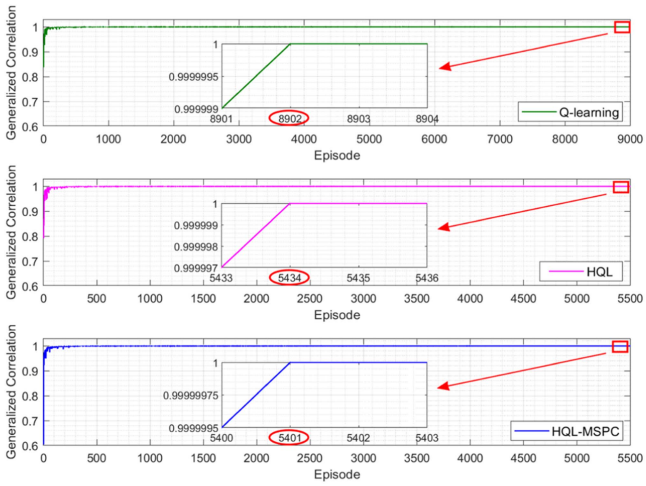


Fig. 13. The variation curves of generalized correlation coefficients for different methods.

HQL-MSPC method in the open area is the fastest compared with other methods, as shown in yellow rectangles. Meanwhile, the autonomous vehicle with HQL-MSPC controller takes the shortest time (221 seconds) to complete the whole motion control task, which proves the rapidity of the proposed method. Besides, Fig. 12 shows the steering angular speeds of four methods. The angular speed of the vehicle with PP-LP controller fluctuates sharply, which is between -30 and 30 degrees. The angular speed of the vehicle with HQL-MSPC controller is in the interval $[-8°, 8°]$, which avoids the large angle steering. Therefore, the driving comfort of the proposed method is guaranteed.

Considering the requirements of practical applications, evaluating the training efficiency of three learning-based methods is necessary. The variation curves of generalized correlation coefficients of the three methods are shown in Fig. 13. Obviously, the correlation coefficients of the three methods converge to 1, which means that the three training processes are finally perfected. Specially, the matrix training of the HQL-MSPC method and HQL method is completed at the 5401th episode and the 5434th episode, respectively. The training speed of these two

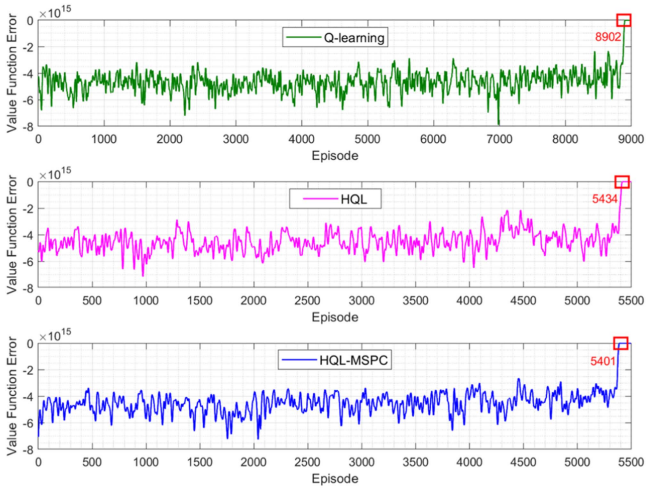


Fig. 14. The variation curves of value function errors for different methods.

TABLE V
THE RESULTS OF PATH TRACKING DERIVED BY FOUR METHODS

Methods	Average Tracking Error (m)	Maximum Tracking Error (m)	Training Time / Calculation Time (s)
PP-LP	1.05	2.35	- / 0.34
Q-learning	0.46	1.16	871.60 / 0.17
HQL	0.43	1.12	532.94 / 0.16
HQL-MSPC	0.23	0.63	529.35 / 0.26

heuristic learning-based methods is much faster than that of the classical Q-learning method, which indicates that the heuristic experience replay enhances the training significantly.

To further evaluate the training efficiency, Fig. 14 shows the variation curves of value function errors of the three methods. As we can see, the errors of the HQL-MSPC and HQL still approach zero faster than classical Q-learning method. Besides, the episode nodes representing the perfect training are consistent with the results in Fig. 13. Therefore, the training effectiveness of HQL-MSPC is proved.

The performance results of path tracking strategies derived by four motion control methods are listed in Table V. The average tracking error of the HQL-MSPC method is the smallest, while the average tracking error of PP-LP method is the largest. The maximum tracking error of the proposed method is also much smaller than errors of other methods. The above comparisons are consistent with the results in Fig. 10, which further proves the best path tracking performance of the HQL-MSPC method. Besides, the training time of proposed method is much shorter than that of the classical Q-learning method, and its calculation time is 0.26 s, which has the potential for the motion control in real-world applications.

Furthermore, the performance results of collision avoidance strategies solved by four motion control methods are listed in Table VI. The extra avoidance distance of HQL-MSPC is the shortest, which implies the vehicle avoids obstacles and continues to track, minimizing the distance traveled. The velocity fluctuation in collision avoidance process of the proposed method

TABLE VI
THE RESULTS OF COLLISION AVOIDANCE DERIVED BY FOUR METHODS

Methods	Extra Avoidance Distance (m)	Velocity Fluctuation (m/s)	Calculation Time (s)
PP-LP	29.97	-	0.57
Q-learning	21.15	1.26	0.16
HQL	20.29	0.72	0.16
HQL-MSPC	17.31	0.61	0.25

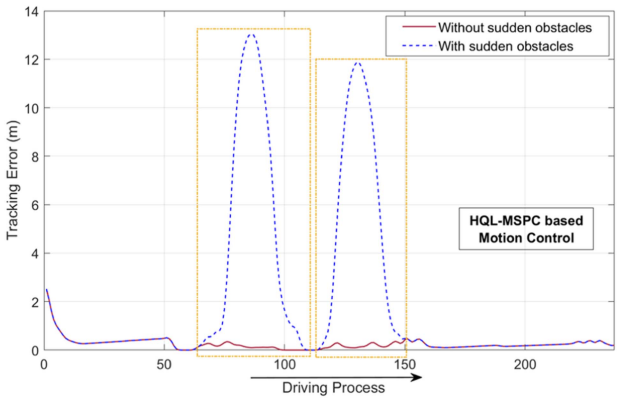


Fig. 15. The tracking error and performance in two comparison scenarios.

is also the least compared with other methods, which ensures the stability of avoidance driving. Besides, its calculation time is 0.25 s, which is also potential for several real-world applications. Therefore, the accuracy, safety, comfort and rapidity of the proposed HQL-MSPC method are demonstrated.

To further demonstrate the flexible adaptation of the proposed HQL-MSPC method in collaborative path tracking and obstacles avoidance, another set of comparison experiment is presented more intuitively. In this experiment, the scenario without sudden obstacles shown in Fig. 6 and the dynamic scenario with obstacles suddenly appearing shown in Fig. 8 are applied as the comparative analysis pairs. The impact of sudden obstacles on the tracking accuracy of the autonomous vehicle will more intuitively show the flexible adaptability of the proposed method. The tracking error variations in these two driving scenarios are shown in Fig. 15. It can be found that the autonomous vehicle controlled by the proposed method can always track the reference path with very low tracking error in the scenario without sudden obstacles. On the other hand, when the autonomous vehicle is driving in the scenario with sudden obstacles, it will stay away from the reference path only in the process of locally avoiding the obstacles, and then quickly return to the reference path after bypassing the obstacles. In the rest part of driving process, the tracking error of the autonomous vehicle is consistent with that in the scenarios without sudden obstacles. The above results and analysis prove that the proposed method can achieve effective collaboration between path tracking and obstacle avoidance. Meanwhile, the proposed method has the flexible adaptation to different scenarios.

TABLE VII
THE RESULTS OF PATH TRACKING FOR ADAPTABILITY VALIDATION

Methods	Average Tracking Error (m)	Maximum Tracking Error (m)	Training Time / Calculation Time (s)
PP-LP	1.01	2.17	- / 0.35
Q-learning	0.43	1.28	728.33 / 0.16
HQL	0.42	1.26	414.26 / 0.17
O-HQL-MSPC	0.24	0.67	- / 0.25
N-HQL-MSPC	0.19	0.59	409.85 / 0.25

The global map shown in Fig. 7 is also updated to the dynamic scenario with obstacles suddenly appearing, and applied for the adaptability verification of proposed motion control method. Specially, two versions of the HQL-MSPC method are used for comparison, which are new version (N-HQL-MSPC) and original version (O-HQL-MSPC). It should be noted that these two versions of the method use the same algorithmic logic, but are trained by different scenario samples.

In the implementation of the N-HQL-MSPC method, the HQL agent is trained by samples from the second scenario shown in Fig. 7. Differently, in the implementation of the O-HQL-MSPC method, the HQL agent is trained by samples from the first scenario shown in Fig. 8. Then, thiese two trained agents are combined with the MSPC respectively and used directly for the motion control solution in the second scenario. The difference in control results between these two methods determines the adaptability of the well-trained Q matrix.

The motion control performance containing path tracking and obstacles avoidance for different methods are illustrated in Fig. 16.

Obviously, the O-HQL-MSPC method still completes path tracking and obstacles avoidance efficiently in the new scenario. The tracking accuracies of the O-HQL-MSPC and the N-HQL-MSPC method are higher than those of the other three methods. Besides, the local collision avoidance trajectories of these two HQL-MSPC methods are smooth enough as shown in blue ellipses.

Table VII shows the performance results of path tracking derived by different methods. The average and maximum tracking errors of the O-HQL-MSPC method are close to those of the N-HQL-MSPC method, and much smaller than those of other methods. In particular, the agent of O-HQL-MSPC method has been trained well in the previous scenario, and it can be applied directly in the new scenario without further training. Its calculation time is 0.25 s, which is possible for real-time applications. Furthermore, Table VIII shows the performance results of collision avoidance derived by different methods. It is observed that the extra avoidance distance and the velocity fluctuation in collision avoidance process of the O-HQL-MSPC method are also close to those of the N-HQL-MSPC method. Its calculation time still satisfies the requirement of the real-time motion control. Finally, the adaptability of the proposed framework in motion control is demonstrated.

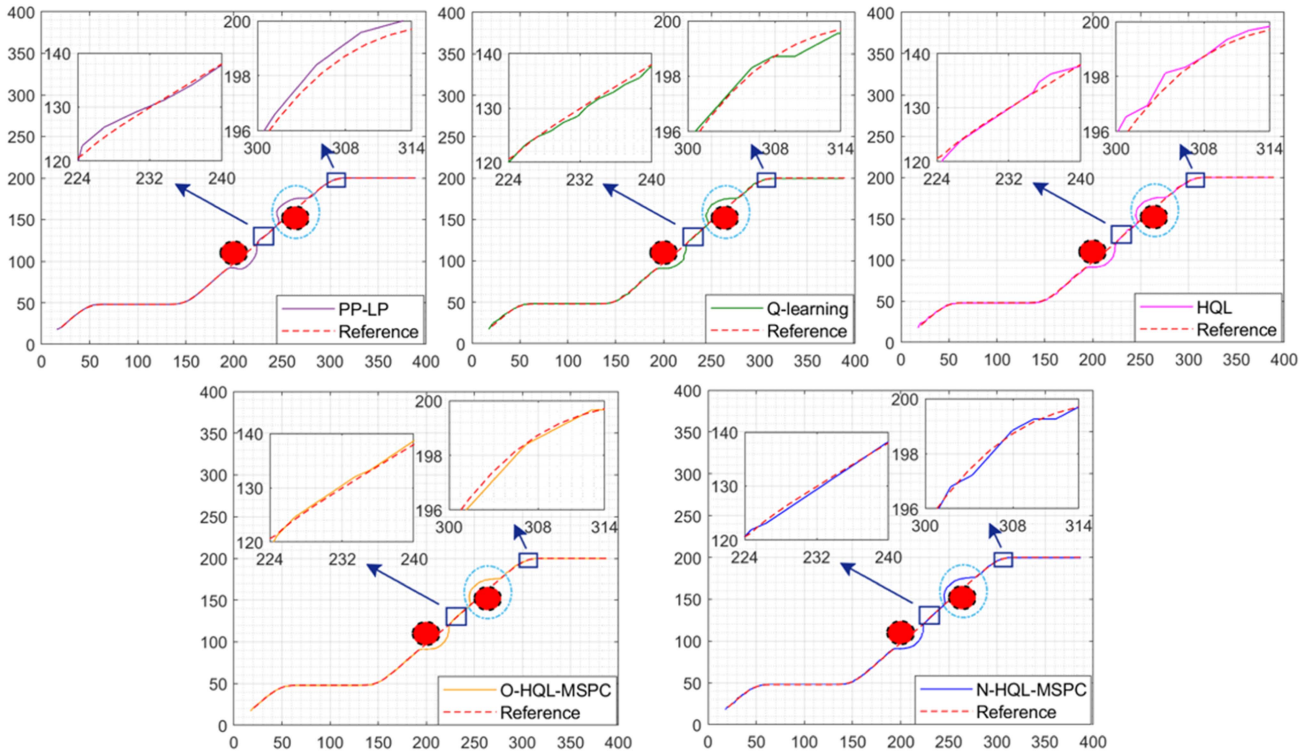


Fig. 16. The motion control performance of these five control methods for adaptability validation.

TABLE VIII
THE RESULTS OF COLLISION AVOIDANCE FOR ADAPTABILITY VALIDATION

Methods	Extra Avoidance Distance (m)	Velocity Fluctuation (m/s)	Calculation Time (s)
PP-LP	34.19	-	0.59
Q-learning	22.06	1.34	0.15
HQL	21.58	0.92	0.16
O-HQL-MSPC	18.37	0.78	0.25
N-HQL-MSPC	17.95	0.60	0.24

C. Validation of Path Planning and Motion Control in the Real World Scenario

To further verify the performance of hierarchical framework, a more complex real world scenario was applied. As mentioned above, the real-world scenario was collected by a real vehicle in the Innovation Center of BIT. Fig. 17 shows the global positioning system (GPS) based satellite map and synchronous positioning and mapping (SLAM) based points cloud map of the collected scenario. For this autonomous driving task, the path planning layer is executed in this fixed scenario and the global path is formed. The autonomous vehicle is required to efficiently track the global path from start position to temporary target position, and then from temporary target position to final target position. Besides, in the process of motion control, several dynamic obstacles are constructed as uncertain disturbance of external environment, which are also illustrated in Fig. 17. When

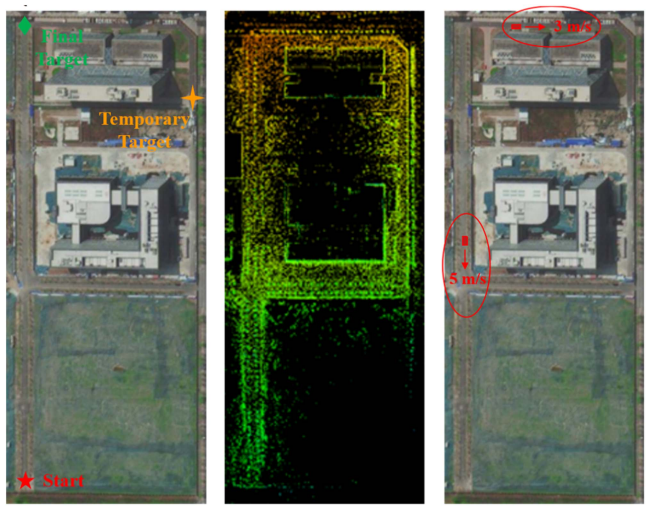


Fig. 17. The satellite map, points cloud map and dynamic obstacles information for real world scenario.

the autonomous vehicle reaches the specific area, the corresponding dynamic obstacles will be triggered to start moving, and the vehicle needs to avoid the obstacles safely and quickly, then return to the reference path to continue driving.

In the evaluation of path planning performance, the Dijkstra algorithm and the fast A$*$ algorithm are used as results comparison with the proposed framework. The global paths planned by different algorithm are shown in Fig. 18. Obviously, these three algorithms successfully complete two stages of path planning

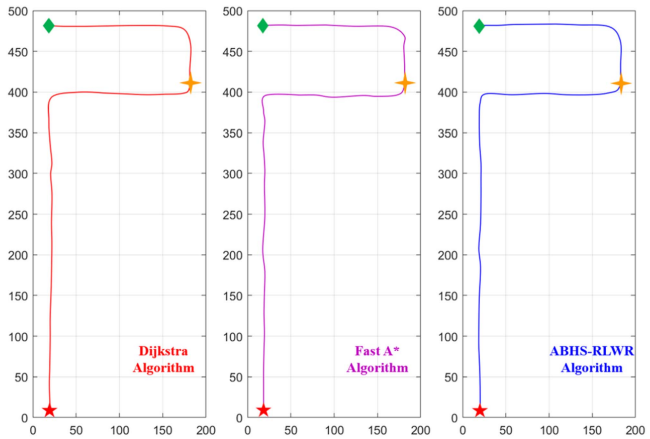


Fig. 18. The global paths of three algorithms for real world scenario.

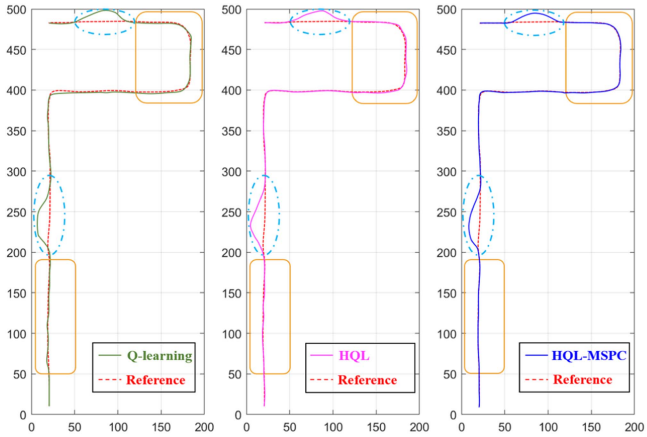


Fig. 19. The motion trajectories controlled by three methods for real world scenario.

TABLE IX
KEY INDICATORS OF PATHS PLANNED AND RELATIVE INCREASES IN REAL WORLD SCENARIO

Algorithms	Total Length (m)	Calculation Time (s)	Maximum Cumulative Curvature
	(Relative Increase (%))		
Dijkstra Algorithm	790.13	22.31	3.95
	(-)	(238.5%)	(51.3%)
Fast A* Algorithm	799.64	8.02	5.36
	(1.2%)	(21.7%)	(105.4%)
ABHS-RLWR Algorithm	792.98	6.59	2.61
	(0.4%)	(-)	(-)

TABLE X
THE RESULTS OF PATH TRACKING GENERATED IN REAL WORLD SCENARIO

Methods	Average Tracking Error (m)	Maximum Tracking Error (m)	Average Driving Velocity (m/s)
Q-learning	0.62	2.17	7.55
HQL	0.51	1.51	7.86
HQL-MSPC	0.18	0.50	9.23

TABLE XI
THE RESULTS OF COLLISION AVOIDANCE GENERATED IN REAL WORLD SCENARIO

Methods	Extra Avoidance Distance (m)	Velocity Fluctuation (m/s)	Calculation Time (s)
Q-learning	35.41	1.49	0.17
HQL	33.79	0.80	0.17
HQL-MSPC	26.58	0.71	0.23

and generate paths with similar trends. Especially, the path of the ABHS-RLWR algorithm is smoother at quarter turn section. Furthermore, Table IX shows the key indicators of paths planned and relative increases in real world scenario. It is evident that the length of ABHS-RLWR algorithm approaches the shortest length from Dijkstra which is benchmark. Meanwhile, the proposed algorithm is much better than Dijkstra algorithm for the index of planning rapidity and smoothness. Besides, the proposed algorithm outperforms the fast A$*$ algorithm in total length, calculation time and maximum cumulative curvature. Therefore, the ABHS-RLWR algorithm is significantly superior to other two algorithms.

In the evaluation of motion control performance, the Q-learning method and the HQL method are used as results comparison with the proposed framework. The path tracking and dynamic collision avoidance trajectories of different methods are illustrated in Fig. 19. As can be seen, the autonomous vehicle equipped with three control methods successfully complete the motion control task, including global path tracking and local obstacle avoidance. From the local motion trajectories in the orange rectangles, it can be found that the proposed HQL-MSPC method has the highest path tracking accuracy, and the tracking performance is excellent in both the straight road section and the turning section. The tracking accuracy of HQL method is better than that of Q-learning method in specific road sections, which also reflects the effectiveness of heuristic experience replay in optimization system. Besides, from the trajectories of the local obstacle avoidance part, it can be found that the vehicle controlled by the HQL-MSPC method can smoothly and quickly complete the dynamic obstacle avoidance and return to the reference path with the shortest extra obstacle avoidance distance.

The results of path tracking generated by three control methods for real world scenario are described in Table X. In numerical terms, the tracking error of the vehicle equipped with the HQL-MSPC method is much smaller than that of the other two methods, and its average driving velocity is the fastest, which indicates that the proposed method is superior to the other two methods in terms of tracking accuracy and rapidity.

Moreover, the results of collision avoidance generated by three control methods for real world scenario are listed in Table XI. The extra avoidance distance of HQL-MSPC method is the shortest, which is consistent with the discussion of Fig. 19.

The proposed method realizes the least extra mileage cost to avoid dynamic obstacles maintaining the fastest average driving velocity. Meanwhile, in the process of collision avoidance, the velocity fluctuation of the proposed method is also smaller than that of other two methods, which ensures the comfort of obstacle-avoiding driving. Therefore, the HQL-MSPC method achieves better effectiveness in accuracy, safety, rapidity and comfort.

Finally, the performance of hierarchical framework is demonstrated in the real world scenario.

V. Conclusion and Research Plan

In this research, the novel hierarchical framework consisting of path planning and motion control of the autonomous vehicle in non-specific scenarios is proposed with several logical associations and algorithmic improvements. In the path planning layer, the bidirectional heuristic planning with adaptive scale search is designed and incorporated with the robust locally weighted regression algorithm to generate the optimal global path. In the motion control layer, the multi-step predictive control method based on the heuristic Q-learning algorithm is proposed to improve the effects of the motion control. The hierarchical framework is proved by the virtual driving environment simulation and real world scenario test. It can be concluded that the proposed framework for autonomous driving achieves better performance in both path planning and motion control compared to several existing algorithms and methods. In the results analysis of path planning part, the rapidity, smoothness and optimality are validated. In the results analysis of motion control part, the safety, accuracy, comfort and rapidity are validated. Furthermore, the adaptability of framework is demonstrated in another autonomous driving scenario.

In the next research plan, the hierarchical framework will be optimized by more non-specific scenarios, then be applied to the real vehicle and real-life experiment.

References

[1] A. Haydari and Y. Yılmaz, "Deep reinforcement learning for intelligent transportation systems: A survey," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 1, pp. 11–32, Jan. 2022.

[2] M. Veres and M. Moussa, "Deep learning for intelligent transportation systems: A survey of emerging trends," *IEEE Trans. Intell. Transp. Syst.*, vol. 21, no. 8, pp. 3152–3168, Aug. 2020.

[3] Y. Huang et al., "A motion planning and tracking framework for autonomous vehicles based on artificial potential field elaborated resistance network approach," *IEEE Trans. Ind. Electron.*, vol. 67, no. 2, pp. 1376–1386, Feb. 2020.

[4] S. Kuutti, R. Bowden, Y. Jin, P. Barber, and S. Fallah, "A survey of deep learning applications to autonomous vehicle control," *IEEE Trans. Intell. Transp. Syst.*, vol. 22, no. 2, pp. 712–733, Feb. 2021.

[5] L. Claussmann, M. Revilloud, D. Gruyer, and S. Glaser, "A review of motion planning for highway autonomous driving," *IEEE Trans. Intell. Transp. Syst.*, vol. 21, no. 5, pp. 1826–1848, May 2020.

[6] C. Yuan, G. Liu, W. Zhang, and X. Pan, "An efficient RRT cache method in dynamic environments for path planning," *Robot. Auton. Syst.*, vol. 131, 2020, Art. no. 103595.

[7] K. Cao, Q. Cheng, S. Gao, Y. Chen, and C. Chen, "Improved PRM for path planning in narrow passages," in *Proc. IEEE Int. Conf. Mechatron. Automat.*, 2019, pp. 45–50.

[8] L. Zhang, Y. Zhang, and Y. Li, "Mobile robot path planning based on improved localized particle swarm optimization," *IEEE Sensors J.*, vol. 21, no. 5, pp. 6962–6972, Mar. 2021.

[9] C. Lamini, S. Benhlima, and A. Elbekri, "Genetic algorithm based approach for autonomous mobile robot path planning," *Procedia Comput. Sci.*, vol. 127, pp. 180–189, 2018.

[10] W. Xu, Q. Wang, and J. M. Dolan, "Autonomous vehicle motion planning via recurrent spline optimization," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2021, pp. 7730–7736.

[11] F. Bounini, D. Gingras, H. Pollart, and D. Gruyer, "Modified artificial potential field method for online path planning applications," in *Proc. IEEE Intell. Veh. Symp.*, 2017, pp. 180–185.

[12] W. Siming, Z. Tiantian, and L. Weijie, "Mobile robot path planning based on improved artificial potential field method," in *Proc. IEEE Int. Conf. Intell. Robotic Control Eng.*, 2018, pp. 29–33.

[13] M. Luo, X. Hou, and J. Yang, "Surface optimal path planning using an extended Dijkstra algorithm," *IEEE Access*, vol. 8, pp. 147827–147838, 2020.

[14] E. Shang, B. Dai, Y. Nie, Q. Zhu, L. Xiao, and D. Zhao, "An improved A-star based path planning algorithm for autonomous land vehicles," *Int. J. Adv. Robot. Syst.*, vol. 17, no. 5, pp. 1–13, 2020.

[15] L. Liu et al., "Global dynamic path planning fusion algorithm combining jump-A$*$ algorithm and dynamic window approach," *IEEE Access*, vol. 9, pp. 19632–19638, 2021.

[16] H. Huang et al., "Dynamic path planning based on improved D$*$ algorithms of Gaode map," in *Proc. IEEE 3rd Inf. Technol., Netw., Electron. Automat. Control Conf.*, 2019, pp. 1121–1124.

[17] I. Maurović, M. Seder, K. Lenac, and I. Petrović, "Path planning for active SLAM based on the D$*$ algorithm with negative edge weights," *IEEE Trans. Syst., Man, Cybern.:, Syst.*, vol. 48, no. 8, pp. 1321–1331, Aug. 2018.

[18] Y. Pan, X. Li, and H. Yu, "Efficient PID tracking control of robotic manipulators driven by compliant actuators," *IEEE Trans. Control Syst. Technol.*, vol. 27, no. 2, pp. 915–922, Mar. 2019.

[19] G. Han, W. Fu, W. Wang, and Z. Wu, "The lateral tracking control for the intelligent vehicle based on adaptive PID neural network," *Sensors*, vol. 17, no. 6, 2017, Art. no. 1244.

[20] W.-J. Wang, T.-M. Hsu, and T.-S. Wu, "The improved pure pursuit algorithm for autonomous driving advanced system," in *Proc. IEEE 10th Int. Workshop Comput. Intell. Appl.*, 2017, pp. 33–38.

[21] Y. Huang, Z. Tian, Q. Jiang, and J. Xu, "Path tracking based on improved pure pursuit model and PID," in *Proc. IEEE 2nd Int. Conf. Civil Aviation Saf. Inf. Technol.*, 2020, pp. 359–364.

[22] S. Xu and H. Peng, "Design, analysis, and experiments of preview path tracking control for autonomous vehicles," *IEEE Trans. Intell. Transp. Syst.*, vol. 21, no. 1, pp. 48–58, Jan. 2020.

[23] F. Eiras, M. Hawasly, S. Albrecht, and S. Ramamoorthy, "A two-stage optimization-based motion planner for safe urban driving," *IEEE Trans. Robot.*, vol. 38, no. 2, pp. 822–834, Apr. 2022.

[24] L. Tang, F. Yan, B. Zou, K. Wang, and C. Lv, "An improved kinematic model predictive control for high-speed path tracking of autonomous vehicles," *IEEE Access*, vol. 8, pp. 51400–51413, 2020.

[25] A. Muraleedharan, H. Okuda, and T. Suzuki, "Real-time implementation of randomized model predictive control for autonomous driving," *IEEE Trans. Intell. Veh.*, vol. 7, no. 1, pp. 11–20, Mar. 2022.

[26] F. Gao, Y. Han, and D. Dang, "Balancing accuracy and efficiency: Fast motion planning based on nonlinear model predictive control," in *Proc. 5th CAA Int. Conf. Veh. Control Intell.*, 2021, pp. 1–6.

[27] Y. Shan, B. Zheng, L. Chen, L. Chen, and D. Chen, "A reinforcement learning-based adaptive path tracking approach for autonomous driving," *IEEE Trans. Veh. Technol.*, vol. 69, no. 10, pp. 10581–10595, Oct. 2020.

[28] V. Behzadan and A. Munir, "Adversarial reinforcement learning framework for benchmarking collision avoidance mechanisms in autonomous vehicles," *IEEE Intell. Transp. Syst. Mag.*, vol. 13, no. 2, pp. 236–241, Summer 2021.

[29] L. Ding, S. Li, H. Gao, C. Chen, and Z. Deng, "Adaptive partial reinforcement learning neural network-based tracking control for wheeled mobile robotic systems," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 50, no. 7, pp. 2512–2523, Jul. 2020.

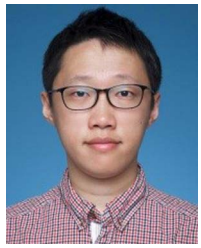
[30] L. Zhang, R. Zhang, T. Wu, R. Weng, M. Han, and Y. Zhao, "Safe reinforcement learning with stability guarantee for motion planning of autonomous vehicles," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 32, no. 12, pp. 5435–5444, Dec. 2021.

[31] X. Wang, H. Krasowski, and M. Althoff, "CommonRoad-RL: A configurable reinforcement learning environment for motion planning of autonomous vehicles," in *Proc. IEEE Int. Intell. Transp. Syst. Conf.*, 2021, pp. 466–472.

[32] S. Aradi, "Survey of deep reinforcement learning for motion planning of autonomous vehicles," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 2, pp. 740–759, Feb. 2022.
[33] Z. He, L. Dong, C. Sun, and J. Wang, "Asynchronous multithreading reinforcement-learning-based path planning and tracking for unmanned underwater vehicle," *IEEE Trans. Syst., Man, Cybern. Syst.*, vol. 52, no. 5, pp. 2757–2769, May 2022, doi: 10.1109/TSMC.2021.3050960.
[34] Y. Lu, X. Xu, X. Zhang, L. Qian, and X. Zhou, "Hierarchical reinforcement learning for autonomous decision making and motion planning of intelligent vehicles," *IEEE Access*, vol. 8, pp. 209776–209789, 2020.
[35] M. Chen et al., "FaSTrack: A modular framework for real-time motion planning and guaranteed safe tracking," *IEEE Trans. Autom. Control*, vol. 66, no. 12, pp. 5861–5876, Dec. 2021.
[36] G. Du, Y. Zou, X. Zhang, L. Guo, and N. Guo, "Energy management for a hybrid electric vehicle based on prioritized deep reinforcement learning framework," *Energy*, vol. 241, 2022, Art. no. 122523.
[37] B. Hu et al., "Reinforcement learning approach to design practical adaptive control for a small-scale intelligent vehicle," *Symmetry*, vol. 11, no. 9, 2019, Art. no. 1139.

Guodong Du received the B.S. degree in mechanical engineering in 2019 from the Beijing Institute of Technology, Beijing, China, where he is currently working toward the Ph.D. degree in automobile engineering. His research interests include autonomous driving, motion planning and control, reinforcement learning algorithm, vehicle dynamics control, and energy management of hybrid electric vehicles. He is an academic guest of ETH Zurich. He was the recipient of the "Best Student Paper Award" in the 35th IEEE Intelligent Vehicles Symposium (IV 2024).

Yuan Zou (Senior Member, IEEE) received the Ph.D. degree from the Beijing Institute of Technology, Beijing, China, in 2005. He is currently a Professor with the Beijing Collaborative and Innovative Center for Electric Vehicles and School of Mechanical Engineering, Beijing Institute of Technology. He is also the Co-Director of ETHZ-BIT Joint Research Center for New Energy Vehicle Dynamic System and Control. He conducted research about ground vehicle propulsion modeling and optimal control with University of Michigan Ann Arbor, Ann Arbor, MI, USA and ETH Zurich, Zürich, Switzerland. His research interests include modeling and control for electrified vehicle, and transportation system.



Xudong Zhang (Member, IEEE) received the M.S. degree in mechanical engineering from the Beijing Institute of Technology, Beijing, China, in 2011 and the Ph.D. degree in mechanical engineering from the Technical University of Berlin, Berlin, Germany, in 2017. Since 2017, he has been an Associate Professor with the Beijing Institute of Technology. His research interests include distributed drive electric vehicles, vehicle dynamics control, vehicle state estimation, torque allocation, and power management of hybrid electric vehicles.

Zirui Li received the B.S. degree in 2019 from the Beijing Institute of Technology, Beijing, China, where he is currently working toward the Ph.D. degree in mechanical engineering under the supervision of Prof. Jianwei Gong. From 2021 to 2022, he was a Visiting Researcher with the Delft University of Technology (TU Delft), Delft, The Netherlands, with CSC funding from China. Since 2022, he has been a Visiting Researcher with the Chair of Traffic Process Automation, Faculty of Transportation and Traffic Sciences "Friedrich List", TU Dresden, Dresden, Germany. His research interests inlcude interactive behavior modeling, risk assessment, and motion planning of automated vehicles.

Qi Liu received the B.S. degree in 2019 from the Beijing Institute of Technology, Beijing, China, where he is currently working toward the Ph.D. degree in mechanical engineering. His research interests include intelligent vehicles, environmental perception, and decision making.