Green Energy and Intelligent Transportation 1 (2022) 100022

Contents lists available at ScienceDirect

Green Energy and Intelligent Transportation

journal homepage: www.journals.elsevier.com/green-energy-and-intelligent-transportation

Full length article

A hybrid motion planning framework for autonomous driving in mixed traffic flow

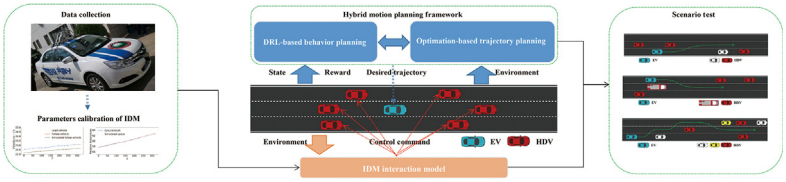
Lei Yang [a], Chao Lu [a], Guangming Xiong [a,*], Yang Xing [b], Jianwei Gong [a]

[a] Intelligent Vehicle Research Center (IVRC), Beijing Institute of Technology, China
[b] Engineering Centre for Autonomous and Cyberphysical Systems, Cranfield University, UK

HIGHLIGHTS

- A motion planning framework is proposed for autonomous driving in mixed traffic flow.
- The bidirectional interaction is considered in behavior planning.
- An intelligent driver model calibrated by real data is used to imitate human drivers.
- An optimization-based trajectory planning is used to improve safety and feasibility.

GRAPHICAL ABSTRACT



ARTICLE INFO

Keywords:
Motion planning
Behavior planning
Trajectory planning
Interaction
Autonomous driving

ABSTRACT

As a core part of an autonomous driving system, motion planning plays an important role in safe driving. However, traditional model- and rule-based methods lack the ability to learn interactively with the environment, and learning-based methods still have problems in terms of reliability. To overcome these problems, a hybrid motion planning framework (HMPF) is proposed to improve the performance of motion planning, which is composed of learning-based behavior planning and optimization-based trajectory planning. The behavior planning module adopts a deep reinforcement learning (DRL) algorithm, which can learn from the interaction between the ego vehicle (EV) and other human-driven vehicles (HDVs), and generate behavior decision commands based on environmental perception information. In particular, the intelligent driver model (IDM) calibrated based on real driving data is used to drive HDVs to imitate human driving behavior and interactive response, so as to simulate the bidirectional interaction between EV and HDVs. Meanwhile, trajectory planning module adopts the optimization method based on road Frenet coordinates, which can generate safe and comfortable desired trajectory while reducing the solution dimension of the problem. In addition, trajectory planning also exists as a safety hard constraint of behavior planning to ensure the feasibility of decision instruction. The experimental results demonstrate the effectiveness and feasibility of the proposed HMPF for autonomous driving motion planning in urban mixed traffic flow scenarios.

1. Introduction

During the last several decades, autonomous driving has experienced vigorous development under the common concern of academia and industry. Autonomous driving is a comprehensive application technology covering computer science, automatic control, vehicle engineering and other frontier subjects, which has great application value for individuals and society. It can not only improve driving safety and reduce traffic accidents, but also relieve traffic pressure and improve traffic efficiency to a certain extent. Most autonomous driving systems are composed of four subsystems: environmental perception, global planning, motion planning and motion control. Among them, the environmental

* Corresponding author.
E-mail address: xiongguangming@bit.edu.cn (G. Xiong).

https://doi.org/10.1016/j.geits.2022.100022
Received 14 June 2022; Received in revised form 2 August 2022; Accepted 10 August 2022
Available online 27 August 2022

perception module provides the surrounding environment information such as passable area, road marking line, dynamic and static obstacles, the global planning mainly provides the reference route to the target location, the motion control module is responsible for calculating specific control quantities such as brake, throttle, and steering wheel angle to drive the autonomous vehicle along the desired trajectory. And the module of motion planning needs to: 1) consider environmental information, global planning information, vehicle kinematic constraint, static/dynamic obstacles constraint and other possible constraints; 2) generate a collision-free, smooth, kinematic and dynamic feasible trajectory to reach the goal configuration while ensuring the safety and comfort of driving; 3) solve the problem within limited runtime and respond to rapid changes in the surrounding environment in real time [1–4]. Recently, autonomous vehicles have been preliminarily applied in relatively simple and closed environments, such as airport shuttles and sightseeing vehicles. However, if autonomous driving is to be widely used in daily life, it is inevitable to face the challenges brought by complex dynamic scenarios. Moreover, it is foreseeable that autonomous vehicles will coexist with HDVs for a long time in the future. And interaction with HDVs is also an important issue for autonomous vehicles. As the main component of the autonomous driving system, motion planning needs to have the ability to generate safe and reliable desired trajectory in the face of the mixed traffic flow.

In the past few decades, many traditional motion planning algorithms have been studied and widely used, which can be mainly divided into four categories: graph search based methods, sampling based methods, interpolation based methods and optimization based methods. Graph search based planners usually use a fixed number of grids or lattices to discrete configuration space, and a directed graph for search has been constructed before the planning [5]. According to the given objective function, different search strategies are used to search the global optimal path in the given directed graph [6–8]. This kind of method is very suitable for the shortest path search between finite nodes or grids, but if the search space is too large, its efficiency is too slow. And the curvature of the trajectory generated by this method is usually discontinuous and not smooth, which is difficult to be used in vehicle tracking. To reduce the search time, stochastic strategies are adopted to generate sampling points for approximating the structure of free space in the sample-based methods [9–11]. It can achieve rapid planning through random searches in free space, but it is difficult to ensure the optimality of the result, and the trajectory is also not smooth. Although the interpolation based method can generate a smoothing trajectory, its optimality is also difficult to be guarantee [12,13]. Besides, this method is very time consuming when dealing with dynamic obstacles. Optimization based planners typically employ high-order curve models or piecewise continuous curves to approximate system dynamics constraints, then solve nonlinear programming problems and generate high-quality smooth paths [14–16]. However, this method is still limited in how to deal with the constraints of obstacles, and the real-time requirements are difficult to guarantee.

In many competitions and projects, these traditional methods can perform well through specific parameter adjustments and combinations of different methods for application scenarios. However, with the continuous development of autonomous driving, the number of scenarios will increase rapidly. This approach is unrealistic, and motion planning algorithms need to have the ability to learn from the environment. Under such circumstances, coupled with the development of computing power, learning-based motion planning methods are proposed [17]. A novel motion planning framework is proposed in Ref. [18], which integrates imitation learning (IL) with traditional sampling-based motion planning algorithms. It adopts IL to learn the driving trajectories of human drivers, so as to achieve the purpose of improving sampling efficiency. The experimental results demonstrate the effectiveness of the method, but the interaction between the ego vehicle and surrounding vehicles is not considered in this framework. A human-like decision making system is proposed to make decision in driving scenes like human drivers [24]. The system uses a convolutional neural network (CNN) model as the perception subsystem to get all the information of the input road scene. Then the specific control commands is generated by the system based on the information obtained from the CNN model. The method can mimic the behavior of human drivers, which can better adapt autonomous vehicles to real traffic scenarios, but it relies heavily on training datasets. Chen et al. [25] propose a new paradigm for vision-based autonomous driving system. A model based CNN is built to realize the mapping from the input image to a small number of key perception indicators, then a simple motion controller is adopted to control the vehicle to drive autonomously. Simulations in virtual and real scenes show that the new method can perform well. But such methods are likely to make unpredictable decisions when faced with unfamiliar scenarios. Recently, reinforcement learning (RL) has been widely used in motion planning due to its advantages in interactive learning [21]. Lu et al. [19] propose a novel automated overtaking system based on hierarchical reinforcement learning (HRL). The social preferences of the vehicles being overtaken are taken into account and extracted using a data-driven approach. The state transition probabilities of overtaken vehicles with different social preferences are obtained by statistical method, and then motion planning and control are carried out based on the probabilities. The test results show that this method can achieve safe and effective overtaking behavior, but the state of the overtaken vehicle in this method is equivalent to fixed and known. Ref. [22] proposes a personalized behavior learning system (PBLS) based on the RL, which can realize human-like longitudinal speed control through the learning from human drivers. And the system can achieve higher driving comfort and smoothness than the traditional cruise control system. However, unstable drivers may lead to high learning errors in system. A decision planning method based on latent space reinforcement learning is presented in Ref. [20] for highway on-ramps scenarios. The combination model of the hidden Markov model and Gaussian mixture regression is used to construct interpretable states, and the latent space model is used to reduce the state dimension. And experimental results demonstrate that the method can achieve merging behavior with a good balance between safety and efficiency. This method simulates traffic uncertainty by setting different speeds and positions of surrounding vehicles, but the surrounding vehicles do not have the ability to interact. In the work carried out by Ref. [23], a new motion planning algorithm composed of traditional motion control methods and deep reinforcement learning (DRL) algorithms. The algorithm adopts the Deep Deterministic Policy Gradient (DDPG) method to train the motion planner given the predefined initial and end states, and generate a trajectory. In order to guarantee the feasibility of the trajectory, a classic tracking control method is employed in the training stage. Although the algorithm can generate a feasible trajectory which meets the demand of motion control, but obstacles is not considered in this work. Ref. [26] combines DDPG with safety-based control methods, where DDPG is used to train control commands for driving along a predefined path without considering other surrounding vehicles, control methods are used for obstacle avoidance and path tracking, then combined with different constants and verified in several simple scenarios. Although these learning-based methods have shown potential in motion planning, many methods only consider the response of the ego vehicle (EV) when dealing with the problem of interaction with surrounding vehicles. Ref. [27] builds an interaction model of surrounding vehicles, but whether the model can reflect the human driver's response is not considered.

In this work, we proposed a novel hybrid motion planning framework (HMPF) consisted of behavior planning and trajectory planning, which is shown in Fig. 1. A deep-reinforcement-learning-based behavior planning method is proposed to deal with the interaction between the EV and other human-driven vehicles (HDVs), and generate the discrete behavior command according to the environmental information, such as lane keeping and lane changing. Moreover, in order to simulate the bidirectional interaction between vehicles in real traffic environment, the intelligent driver model (IDM) [28] is employed to drive the HDVs in this work. And the parameters of IDM are obtained by calibration from the

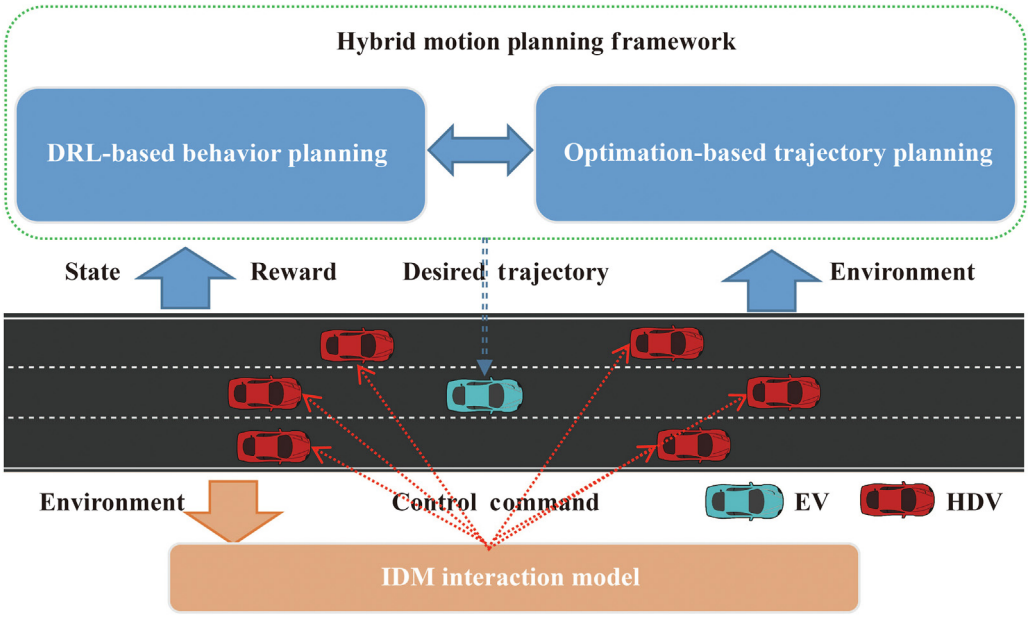


Fig. 1. Hybrid motion planning framework.

collected driving data. Trajectory planning adopts an optimization method based on the frenet coordinate system, which can improve the solution efficiency through horizontal and vertical decoupling, and can take driving comfort into consideration. At the same time, the traditional optimization-based method can also serve as a hard constraint of HMPF to ensure security.

The main contributions of this paper are as follows:

- We propose a hybrid motion planning framework consisting of learning-based behavior planning and optimization-based trajectory planning, which is capable of learning while still maintaining the safety and reliability of traditional methods;
- For mixed traffic flow scenarios where the EV and HDVs coexist, the bidirectional interaction between them is considered in the behavior planning;
- The IDM model with parameter calibration based on real driving data is used to drive HDVs, so that the interaction reaction of HDVs is closer to that of human drivers.

The rest of this paper is organized as follows. Section 2 shows the problem formulation and the hybrid motion planning framework. Section 3 introduces the experiments and the results for the proposed method. Finally, conclusions are drawn in Section 4.

2. Material and methods

2.1. Interaction model

As mentioned in Section 1, many studies only consider the response of the EV when dealing with the interaction between the EV and HDVs. Specifically, these studies directly or indirectly assume that the HDVs are driving in a predetermined manner, or that the EV is able to control the HDVs. The interaction considered in this work is the bidirectional interaction between EV and HDVs, and only the longitudinal interaction of HDVs is considered. The reinforcement learning is employed to build the interaction model of the EV to react to the driving behavior of HDVs, and the IDM is adopted to drive the HDVs, it can also respond to the EV's driving behavior. The IDM model enables HDVs to adjust their speed according to the driving information of the front vehicle. Particularly, real driving data is used to calibrate IDM parameters so that the HDVs can behave more like a human driver.

2.1.1. Data collection

We collect the real driving data by BYD Surui, an autonomous platform of Intelligent Vehicle Research Center (IVRC) of Beijing Institute of Technology (BIT), which is shown as Fig. 2. The data acquisition platform was equipped with on-board sensors to collect vehicle and traffic data. An inertial measurement unit (IMU) and a GPS are used to collect the position information such as latitude and longitude. The 32-wire liDAR is mainly responsible for obtaining environmental information and surrounding vehicle information, while the single-wire radar and millimeter wave radar are used to obtain information of the vehicle ahead. The camera is mainly used to collect image information. The vehicle speed, throttle, brake and other information can be obtained through the vehicle CAN network.

The data acquisition vehicle is driven by a human driver with more than ten years of driving experience.

Fig. 2. Data collection platform.

2.1.2. Parameters calibration of IDM

IDM is a longitudinal car-following model for the traffic simulation of urban and freeway environments, which can ensures no accidents and covers a wide range of scenarios, including cruising in free-traffic scenario, car-following scenario, stopping scenario. The definition is as follows:

$$\dot{x} = \frac{\mathrm{d}x}{\mathrm{d}t} = v \tag{1}$$

$$\dot{v} = \frac{\mathrm{d}v}{\mathrm{d}t} = a\left(1 - \left(\frac{v}{v_0}\right)^{\delta} - \left(\frac{s^*(v,\Delta v)}{s}\right)^2\right) \tag{2}$$

$$s^*(v,\Delta v) = s_0 + \max\left(0, vT + \frac{V\Delta V}{2\sqrt{ab}}\right) \tag{3}$$

IDM has six parameters, as shown in Table 1, where v_0 presents the desired velocity that the vehicle can achieve under the conditions permitted by the traffic environment and regulations. s_0 presents the minimum desired distance when the vehicle can not move ahead. T is the desired time headway, which is the minimum time to the vehicle in front. a is the maximum vehicle acceleration. b is the comfortable braking deceleration, which is a positive number. And exponent δ is usually set to 4 [29]. Therefore, five parameters need to be calibrated, which can be represented by $P(v_0, s_0, T, a, b)$.

Table 1
IDM parameters.

Notation	Name (units)
v_0	Desired maximum velocity ($\mathrm{m{\cdot}s^{-1}}$)
s_0	Desired minimum distance (m)
T	Desired time headway (s)
a	Maximum vehicle acceleration ($\mathrm{m{\cdot}s^{-2}}$)
b	Comfortable braking deceleratioin ($\mathrm{m{\cdot}s^{-2}}$)
δ	Exponent

In this work, the nonlinear optimization [30] is used to get the optimal model parameters $P^*(v_0^*, s_0^*, T^*, a^*, b^*)$. Define a finite numerical update time Δt, and the IDM simulation process can be described by Eqs. (4)-(7),

$$v_{\mathrm{idm}}(t+\Delta t) = v_{\mathrm{idm}}(t) + \dot{v}_{\mathrm{idm}}(t)\Delta t \tag{4}$$

$$x_{\mathrm{idm}}(t+\Delta t) = x_{\mathrm{idm}}(t) + v_{\mathrm{idm}}(t)\Delta t + \frac{1}{2}\dot{v}_{\mathrm{idm}}(t)\Delta t^2 \tag{5}$$

$$s_{\mathrm{idm}}(t+\Delta t) = x_{\mathrm{data}}^{\mathrm{lead}}(t+\Delta t) - x_{\mathrm{idm}}(t+\Delta t) - L_{\mathrm{lead}} \tag{6}$$

$$\dot{v}_{\mathrm{idm}}(t) = a\left[1 - \left(\frac{v_{\mathrm{idm}}(t)}{v_0}\right)^4 - \left(\frac{s_0 + \max\left(0, Tv_{\mathrm{idm}}(t) + \frac{v_{\mathrm{idm}}(t)\left(v_{\mathrm{idm}}(t) - v_{\mathrm{data}}^{\mathrm{lead}}(t)\right)}{2\sqrt{ab}}\right)}{s_{\mathrm{idm}}(t)}\right)^2\right] \tag{7}$$

and the initialization is as follows:

$$v_{\mathrm{idm}}(t=0) = v_{\mathrm{data}}(0) \tag{8}$$

$$s_{\mathrm{idm}}(t=0) = s_{\mathrm{data}}(0) \tag{9}$$

where v_{idm} and x_{idm} are the velocity and position obtained by IDM simulation, $x_{\mathrm{data}}^{\mathrm{lead}}$ and $v_{\mathrm{data}}^{\mathrm{lead}}$ are the position and velocity of the leading vehicle in the collected data, L_{lead} is its length. And s_{data} represents the relative distance between the leading and following vehicles in the data. The deviation between simulation and ground truth is formulated as Eq. (10).

$$D_s = |s_{\mathrm{idm}} - s_{\mathrm{data}}|, D_v = |v_{\mathrm{idm}} - v_{\mathrm{data}}| \tag{10}$$

In this work, relative distance deviation D_s is optimized, and the objective function is shown as follows:

$$\begin{aligned} \textbf{minimize} \quad & \mathcal{J} =: \frac{1}{T_e}\int_0^{T_e} D_s(\tau)^2 \mathrm{d}(\tau) \\ \text{s.t.} \quad & 0.1 \le a \le 5, \\ & 0.1 \le b \le 5, \\ & 0.1 \le s_0 \le 10, \\ & 0.1 \le T \le 5, \\ & 1 \le v_0 \le 30, \end{aligned} \tag{11}$$

The result of calibration is shown in Fig. 3 and Table 2. As shown in Fig. 3(a), the green curve represents the speed profile of the following vehicle simulated by IDM, the red curve is the speed profile of the real following vehicle in collected data, and Fig. 3(b) represents the deviation between simulated speed and real speed. Fig. 3(c) and (d) show the deviation between the simulated relative distance of leading and following vehicles obtained by calibrated IDM and the ground truth. The results show that the calibrated IDM model can simulate the driving behavior of human drivers well, and can be used to drive the surrounding vehicles in the simulation environment to simulate the interaction between HDVs and EV.

2.2. Behavior planning

Behavior planning, also known as behavioral decision making in many relevant studies, is mainly responsible for generating discrete driving behavior commands based on the behavior of other traffic participants,

Table 2
Calibration error.

Notation	MAE	MRE	RMSE	Units
Velocity	0.051,7	0.000,8	0.068,8	–
Relative distance	0.019,6	0.000,8	0.022,9	–

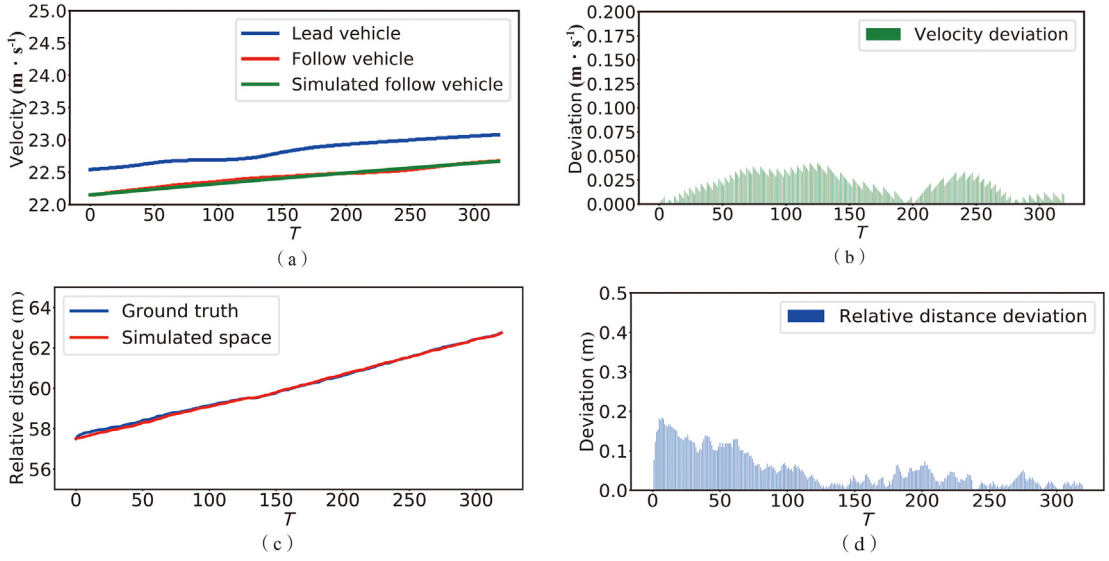


Fig. 3. The results of calibration for IDM parameters.

road conditions and environment information. For example, when the EV finds that the vehicle in front of the lane is slow and there is no vehicle in the adjacent lane, the behavior planning layer will order the autonomous vehicle to change lanes. The method of finite state machines is adopted for behavior planning by most teams in the DARPA Urban Challenge [31], but it is difficult to cover all scenarios with such a rules-based approach. In this work, a DRL-based method is employed for behavior planning.

2.2.1. Reinforcement learning

Reinforcement learning is a learning method in which an agent maximizes the reward signal while implementing a state-to-action mapping [32]. The basic idea of RL is to obtain strategies that can generate optimal action sequences in the process of continuous interaction between the agent and the environment. Unlike other learning methods such as imitation learning, the training data in reinforcement learning is obtained in the process of continuous interaction between the agent and the environment, rather than obtaining a large amount of data before starting training. RL problems are usually modeled as Markov decision processes (MDP), represented by tuple ($\boldsymbol{S, A, R, P,}$ γ). Among them, $\boldsymbol{S}$ is the state space, which represents the state set describing the dynamic environment; $\boldsymbol{A}$ is the action space, which represents all available actions that the agent can choose; $\boldsymbol{R}$ is the reward matrix, which represents the reward obtained by the agent through taking a certain action; $\boldsymbol{P}(s'|s, a)$ is the state transition probability matrix, which describes the transition between states; $\gamma \in [0, 1]$ is the discount factor, which is used to control the importance relationship between future rewards and immediate rewards. The overall goal of the agent is to determine the optimal strategy π^* by maximizing the expected cumulative reward, so as to choose the optimal action with the highest expected cumulative reward in each state, as shown in the following formula:

$$\pi^* = \arg\max_{\pi} E\left[\sum_{t=0}^{\infty} \gamma^t R(S_t, A_t)\right] \tag{12}$$

A. State space

The state space is used to describe the state of the agent and the external environment at any time, which not only needs to grasp the key characteristics of the specific problem objectively and accurately, but also needs to meet certain generalization requirements. During behavior planning, autonomous vehicles need to consider the driving state of themselves and other vehicles around them, so as to make correct decision instructions. In this work, the state space of RL agent is the state of ego vehicle and surrounding vehicles. As shown in Fig. 4, the surrounding vehicles are the front vehicle, rear vehicle, left front vehicle, left rear vehicle, right front vehicle and right rear vehicle. And the state information includes the vehicle's position and velocity over the past five moments, which is defined by Eq. (13), where I^t_{ego} represents the state information of the EV, I^t_{fv} represents the information of the front vehicle, I^t_{rv} represents the information of the rear vehicle, I^t_{lfv} represents the information of the left front vehicle, I^t_{lrv} represents the information of the left rear vehicle, I^t_{rfv} represents the information of the right front vehicle, I^t_{rrv} represents the information of the right rear vehicle.

$$\boldsymbol{S} = \left\{I^t_{\text{ego}}, I^t_{\text{fv}}, I^t_{\text{rv}}, I^t_{\text{lfv}}, I^t_{\text{lrv}}, I^t_{\text{rfv}}, I^t_{\text{rrv}}\right\} \tag{13}$$

B. Action space

The RL-based behavior planning agent plays the role of decision making in HMPF, and it is mainly responsible for generating discrete behavior commands based on environmental information. Analogous to the basic driving behavior of a human driver, the action space is defined as follows:

$$\boldsymbol{A} = \{CL, LK, CR\} \tag{14}$$

where CL represents the behavior of changing lane to the left, LK is lane keeping, and CR represents the behavior of changing lane to the right.

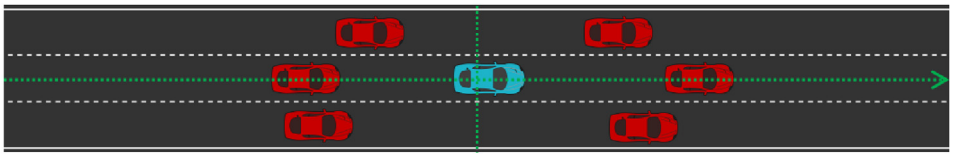


Fig. 4. State space.

C. Reward

Generating a collision-free desired trajectory is critical for motion planning, and efficiency is also a very important factor. The reward is defined by Eq. (15), where the R_{col} represents the collision penalty (set as -10), R_{lc} is the lane change reward (set as 5). R_{eff} is the efficiency reward, K_{v} is a constant coefficient, R_{v} is the velocity of EV, R_{sta} is a constant penalty value used to prevent vehicles from standing still. And R_{tp} is the reward from trajectory planning (TP), which will be a negative value when the desired trajectory corresponding to the current action cannot be generated.

$$R = R_{\text{col}} + R_{\text{lc}} + R_{\text{eff}} + R_{\text{tp}} \tag{15}$$

$$R_{\text{eff}} = K_{\text{v}} R_{\text{v}} + R_{\text{sta}} \tag{16}$$

$$R_{\text{tp}} = \begin{cases} -10 & \text{if TP no solution} \\ 0 & \text{else} \end{cases} \tag{17}$$

D. Training process

At the beginning of each episode, EV will start from any position, and 5–10 HDVs driven by IDM will be randomly placed. The behavior planning module will generate an action according to the observed state information, and the trajectory planning module will try to map the action to the desired trajectory according to the vehicle state and environmental information. The result of trajectory planning will be fed back to the behavior planning module as part of the action reward. The train result is shown in Fig. 5.

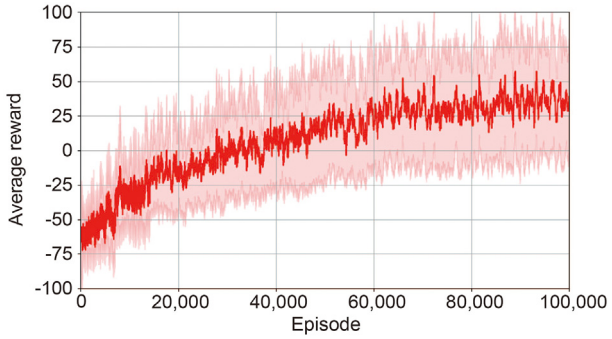


Fig. 5. Train result.

2.3. Trajectory planning

In the autonomous driving system, trajectory planning is responsible for generating the driving trajectory of the vehicle in the future, and the quality of the trajectory is directly related to the driving safety and stability of the vehicle. In this work, a frenet-based trajectory planning is adopted to generate desired trajectory according to the environment information and action command from behavior planning. Meanwhile, the result of trajectory planning will also be used as feedback to evaluate the feasibility of the behavior planning action command.

2.3.1. Frenet coordinate system

For driving in urban environment, human drivers usually take lane centerline or road boundary as reference to plan future driving path. This kind of street-related reference system is usually called Frenet coordinate system, which is shown in Fig. 6. And trajectory planning based on Frenet coordinate system is more conducive to generate the desired trajectory of autonomous vehicles with human-like driving behaviors [33]. As shown in Fig. 6, the lateral offset to the reference path (black dotted curve) is denoted by $d(t)$, the longitudinal covered arc length from the frame's root point s_o is denoted by $s(t)$. The point E on the trajectory of the vehicle should be expressed as $E = (s, d)$ in the Frenet coordinate system. The motion of the vehicle is decomposed into two spaces, longitudinal $s(t)$ and lateral $d(t)$, which is easier to solve.

In trajectory planning, polynomial curves are often used to describe trajectory, including cubic polynomial, quintic polynomial and seventh degree polynomial. However, the cubic polynomial can not guarantee the continuity of acceleration, while the seventh polynomial may have the uncertainty of solving time, so the quintic polynomial is usually used to solve the trajectory planning. A quintic polynomial is shown as Eq. (18).

$$f(t) = p_0 + p_1 t + p_2 t^2 + p_3 t^3 + p_4 t^4 + p_5 t^5 \tag{18}$$

Given the initial state F_I and the goal state F_G, where

$$\begin{aligned} F_I &= \left[f(t_i), \dot{f}(t_i), \ddot{f}(t_i)\right] \\ F_G &= \left[f(t_g), \dot{f}(t_g), \ddot{f}(t_g)\right] \end{aligned} \tag{19}$$

The polynomial equation can be described as Eq. (20),

$$\begin{aligned} f(t_i) &= p_0 + p_1 t_i + p_2 t_i^2 + p_3 t_i^3 + p_4 t_i^4 + p_5 t_i^5 \\ \dot{f}(t_i) &= p_1 + 2p_2 t_i + 3p_3 t_i^2 + 4p_4 t_i^3 + 5p_5 t_i^4 \\ \ddot{f}(t_i) &= 2p_2 + 6p_3 t_i + 12p_4 t_i^2 + 20p_5 t_i^3 \\ f(t_g) &= p_0 + p_1 t_g + p_2 t_g^2 + p_3 t_g^3 + p_4 t_g^4 + p_5 t_g^5 \\ \dot{f}(t_g) &= p_1 + 2p_2 t_g + 3p_3 t_g^2 + 4p_4 t_g^3 + 5p_5 t_g^4 \\ \ddot{f}(t_g) &= 2p_2 + 6p_3 t_g + 12p_4 t_g^2 + 20p_5 t_g^3 \end{aligned} \tag{20}$$

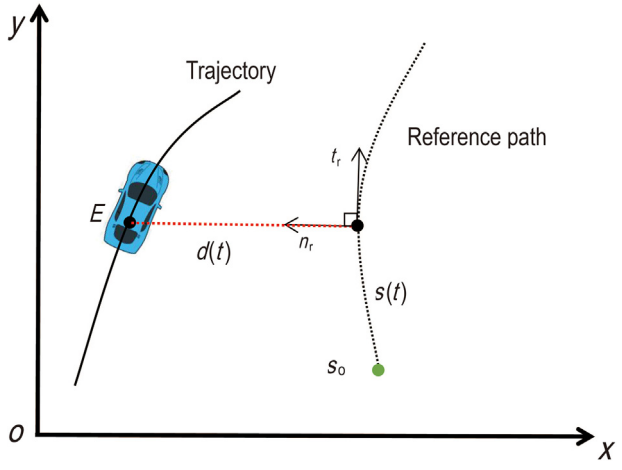


Fig. 6. Vehicle trajectory representation in the Frenet coordinate.

which can also be expressed in Eq. (21).

$$\begin{bmatrix} 1 & t_i & t_i^2 & t_i^3 & t_i^4 & t_i^5 \\ 0 & 1 & 2t_i & 3t_i^2 & 4t_i^3 & 5t_i^4 \\ 0 & 0 & 2 & 6t_i & 12t_i^2 & 20t_i^3 \\ 1 & t_g & t_g^2 & t_g^3 & t_g^4 & t_g^5 \\ 0 & 1 & 2t_g & 3t_g^2 & 4t_g^3 & 5t_g^4 \\ 0 & 0 & 2 & 6t_g & 12t_g^2 & 20t_g^3 \end{bmatrix} \begin{bmatrix} p_0 \\ p_1 \\ p_2 \\ p_3 \\ p_4 \\ p_5 \end{bmatrix} = \boldsymbol{TP} = \begin{bmatrix} F_I & F_G \end{bmatrix}^T \tag{21}$$

The parameters of Eq. (18) can be solved by Eq. (22).

$$\boldsymbol{P} = \boldsymbol{T}^{-1} \begin{bmatrix} F_I & F_G \end{bmatrix}^T \tag{22}$$

Moreover, in order to ensure the comfort of the trajectory, the method of minimizing **jerk** is adopted for objective optimization [33,34], as shown in Eq. (23).

$$J(f(t)) := \int_{t_i}^{t_g} \dddot{f}(\tau)^2 d(\tau) \tag{23}$$

2.3.2. Lateral trajectory

For lateral planning, given the initial state $D_I = \left[d(t_i), \dot{d}(t_i), \ddot{d}(t_i)\right]$ and the goal state $D_G = \left[d(t_g), \dot{d}(t_g), \ddot{d}(t_g)\right]$, the lateral quintic polynomial $d(t)$ can be solved by Eq. (22), and the lateral cost function C_{LAT} is defined as Eq. (24), where T_D represents the time taken from the initial state to the goal state, D_D represents the deviation between the goal position and the reference path, K_J, K_T and K_D are constant coefficients. By adjusting the goal state D_G, the lateral trajectory set can be obtained, which is shown in Fig. 7.

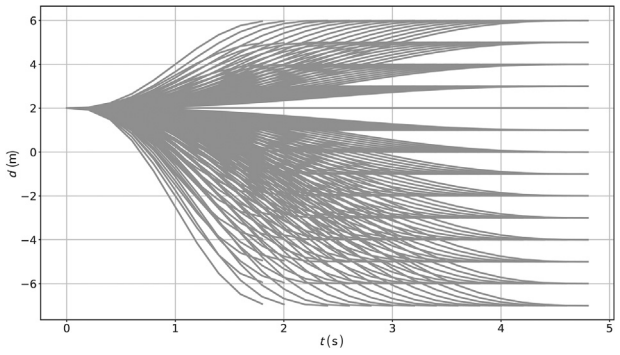


Fig. 7. The result of lateral planning.

$$C_{LAT} = K_J J(d(t)) + K_T T_D + K_D |D_D| \tag{24}$$

2.3.3. Longitudinal trajectory

For longitudinal trajectory planning, specific driving modes need to be generated first according to the result of behavior planning. There are four driving modes in this work, as shown in Eq. (25), and the algorithm for driving mode generation is shown in Algorithm 1, where the value of F_{stop} is determined by a separate judgment module, which will obtain information such as traffic lights, traffic signs, and route end points in real time, then judges whether to stop based on this information. For the surrounding vehicles, we assume that they move at a constant speed in the planning horizon.

$$M = \{VelocityKeeping, Following, Merging, Stopping\} \tag{25}$$

For driving modes such as *Following*, *Merging* and *Stopping* that require a goal position, given the initial state $S_I = \left[s(t_i), \dot{s}(t_i), \ddot{s}(t_i)\right]$ and the goal state $S_G = \left[s(t_g), \dot{s}(t_g), \ddot{s}(t_g)\right]$, a quintic polynomial $s(t)$ can be obtained by

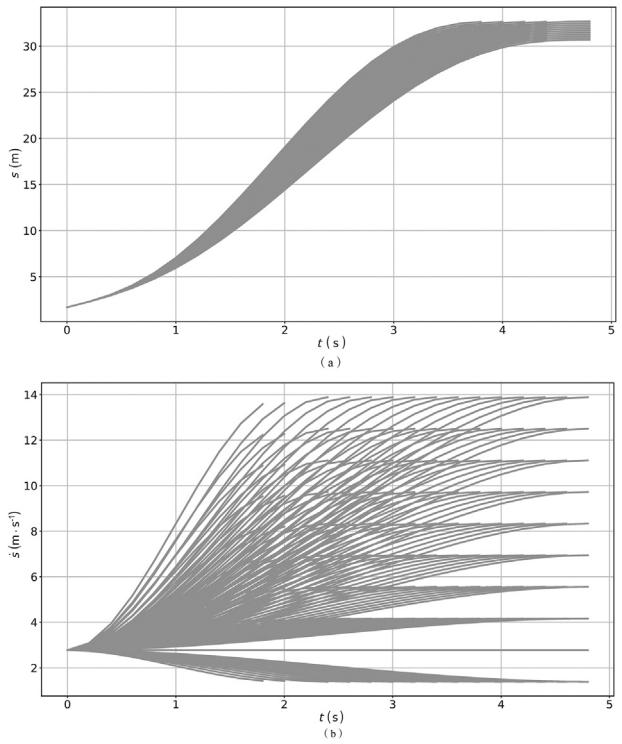


Fig. 8. The result of longitudinal planning.

where s_{lv} represents the position of the leading vehicle, d_0 is the safety distance, k_0 is a constant parameter. For the *Merging* mode, $s(t_g) = (s_{fv} + s_{rv})/2$, among them, s_{fv} and s_{rv} represent the positions of the front and rear vehicles on the target lane, respectively. And for the *Stopping* mode, $s(t_g) = s_{stop}$, where s_{stop} can be the position of the stop line. By adjusting the goal state S_G, a set of longitudinal trajectories can be obtained, as shown in Fig. 8(a)

$$C_{LON} = K_J J(s(t)) + K_T T_S + K_D |D_S| \tag{26}$$

It is worth noting that the longitudinal goal position does not need to be given for the driving mode of *VelocityKeeping*, and the longitudinal trajectory $s(t)$ can be described by a quartic polynomial, as shown in Eq. (27).

$$s(t) = s_0 + s_1 t + s_2 t^2 + s_3 t^3 + s_4 t^4 \tag{27}$$

Given the initial state $S_I = [s(t_i), \dot{s}(t_i), \ddot{s}(t_i)]$ and the goal state $S_G = [\dot{s}(t_g), \ddot{s}(t_g)]$, $s(t)$ can be solved by Eq. (28). At this time, the cost function C_{LON} is defined by Eq. (29). Fig. 8(b) shows the generated longitudinal trajectories.

$$\begin{aligned} s(t_i) &= s_0 + s_1 t_i + s_2 t_i^2 + s_3 t_i^3 + s_4 t_i^4 \\ \dot{s}(t_i) &= s_1 + 2s_2 t_i + 3s_3 t_i^2 + 4s_4 t_i^3 \\ \ddot{s}(t_i) &= 2s_2 + 6s_3 t_i + 12s_4 t_i^2 \\ \dot{s}(t_g) &= s_1 + 2s_2 t_g + 3s_3 t_g^2 + 4s_4 t_g^3 \\ \ddot{s}(t_g) &= 2s_2 + 6s_3 t_g + 12s_4 t_g^2 \end{aligned} \tag{28}$$

$$C_{LOG} = K_J J(s(t)) + K_T T_V + K_D |D_V| \tag{29}$$

Algorithm 1 Drive Mode Generation

Require: The set of HDVs information S_{HDV}; The EV information S_{EV}; The output of behavior planning A_{BP}.
Ensure: M

```
M = VelocityKeeping; // Initial driving mode.
F_stop = False; // The flag of Stopping mode.
L_cur = GetCurrentLane(S_EV); // Get current lane.
L_goal = GetGoalLane(S_EV, A_BP); // Get goal lane.
S_lane = GetVehicleInfo(S_EV, L_goal); // Get vehicle information in goal lane, order from front to back.
if F_stop then
    M = Stopping;
else
    if A_BP == LC then
        if (GetRelativeDis(S_lane[begin], S_EV) < 0) then
            M = VelocityKeeping;
        else
            M = Following;
    else
        if GetRelativeDis(S_lane[begin], S_EV) < 0  AND  S_lane[begin].speed < S_EV.speed then
            M = VelocityKeeping;
        else if S_lane[end].speed > S_EV.speed  OR  GetRelativeDis(S_lane[end], S_EV) > 0 then
            M = Following;
        else
            for i=begin to end do
                if GetFreeSpace(S_lane[i], S_lane[i + 1], S_EV) > MinMergeSpace then
                    M = Merging;
return M;
```

Eq. (22), and the longitudinal cost function C_{LON} is shown in Eq. (26). In this work, the calculation method of the target point is determined by the driving mode [34]. For the *Following* mode, $s(t_g) = s_{lv} - (d_0 + k_0 \dot{s}_{lv})$,

2.3.4. Trajectory selection

As mentioned above, collision-free is the first guarantee of trajectory planning. Therefore, security check is essential for candidate trajectory

set generated by lateral and longitudinal planning. The friction circle constraint [35] is a dynamic hard constraint that the trajectory needs to meet, which can be checked by the acceleration constraint, while the nonholonomic motion constraint can be checked by the curvature constraint. At the same time, a two-phase collision checking method [36] is used to carry out the trajectory collision detection. The final desired trajectory is selected by comparing the cost values of those trajectories that pass the safety check, and the cost function is defined by Eq. (30), where K_{LAT} and K_{LON} are constant coefficients. As shown in Fig. 9, the blue dot is the EV position, the blue curve is the reference line, and the rest curves are the set of all candidate trajectories generated. Among them, the yellow curve represents the trajectory that passes the security check, and the green curve connected by the dot is the desired trajectory finally selected.

$$C_T = K_{\mathrm{LAT}} C_{\mathrm{LAT}} + K_{\mathrm{LON}} C_{\mathrm{LOG}} \tag{30}$$

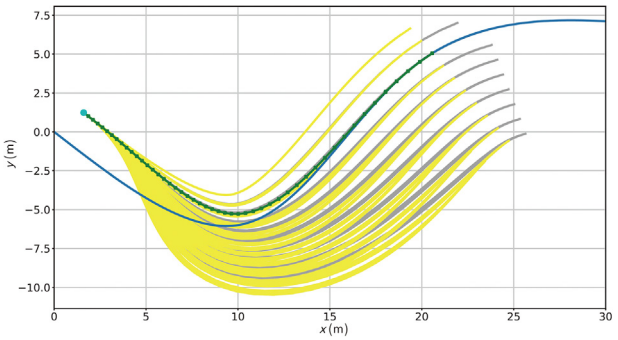


Fig. 9. The result of frenet based trajectory planning.

3. Results and discussion

To ensure the longitudinal interaction capability of HDVs driven by IDM model, we conduct verification tests in the scenario shown in Fig. 10. In Fig. 10, the front vehicle is set to drive at a constant speed of 12 m/s, the rear vehicle is driven by the IDM model, and the maximum speed is set to 20 m/s. Fig. 11 shows the test results, in which the red curve is the driving information of the rear vehicle, and the green curve is the driving information of the front vehicle. As shown in the speed profile in Fig. 11, the rear vehicle driven by the IDM model has the longitudinal interaction ability to adjust the speed autonomously.

The proposed HMPF is tested in three different simulation scenarios, as shown in Fig. 12. Among them, Fig. 12(a) is the scenario of changing lane to the left, Fig. 12(b) is the scenario of changing lane to the right, and Fig. 12(c) is the continuous lane change scenario. Fig. 12(a) and (b) are typical scenarios with high frequency in daily traffic environment, while Fig. 12(c) has a higher complexity. These three scenarios can verify the reliability and adaptability of the HMPF. The algorithm framework is implemented based on Python, and CARLA [37] simulation software is used for experimental testing. The running cycle of the algorithm is 10 Hz, which can meet the real-time requirements. In the trajectory planning, the sampling range of lateral planning is determined by the lane width and the result of behavior planning. In this work, the lane width is set to 3.5 m, the sampling range is determined according to the goal lane, and the sampling interval is 1 m. The sampling range of longitudinal planning is fixed, and the sampling interval of speed is 1.4 m/s. The time range is fixed from 1.8 s to 4.8 s, and the sampling interval of time is 0.2 s.

3.1. Testing scenario 1

The testing one is the scenario of changing lane to the left, which is shown as Fig. 12(a). The EV drives along the right-most lane, and the road speed limit is 80 km/h. After the vehicle travels from the initial

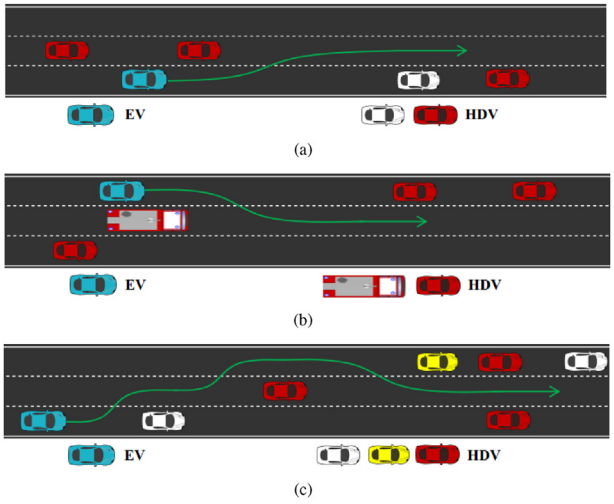


Fig. 12. Test scenarios of EV.

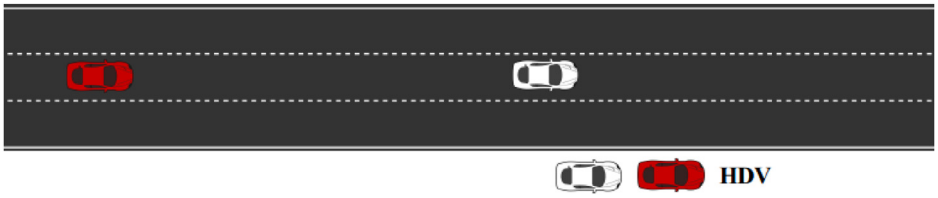


Fig. 10. The test scenario of HDVs.

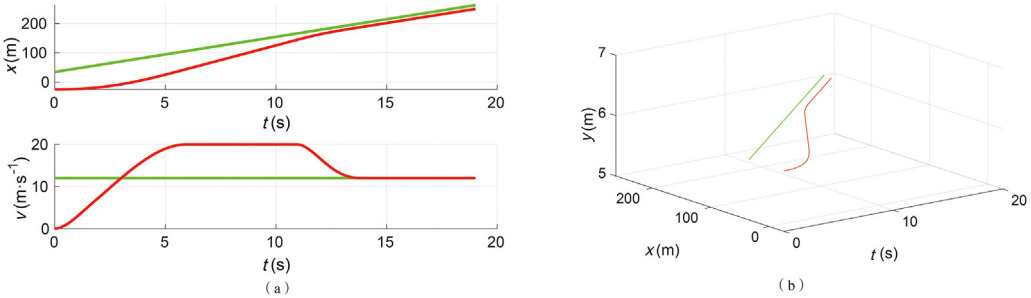


Fig. 11. The test results for longitudinal interaction of HDVs.

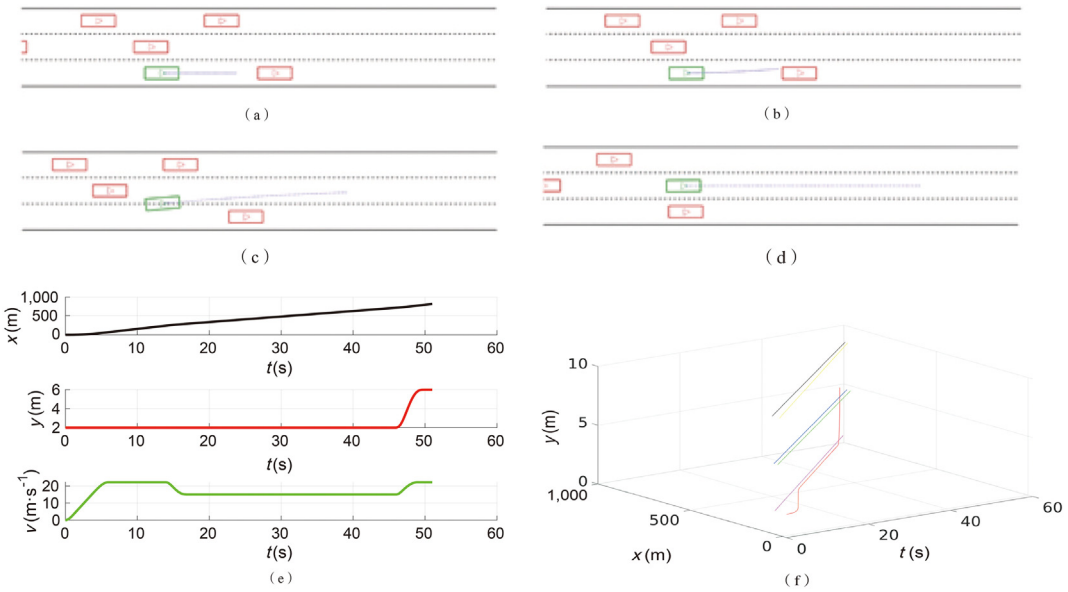


Fig. 13. Changing lane to the left.

position to the speed limit value, it enters the mode of *VelocityKeeping*, and still drives along the right-most lane. When the EV finds that the speed of the vehicle ahead is very slow and the left lane is occupied by other vehicles and cannot change lanes, it will slow down and enter the *Following* mode. And when the behavior planning generates the command of CL, and the desired trajectory generated by trajectory planning is safe and collision-free, the EV performs the CL behavior and enters the *VelocityKeeping* mode. Fig. 13(a)-(d) show some moments in the driving process, in which the green rectangle represents EV, the red rectangle represents HDVs, the blue dotted line is the desired trajectory generated by the trajectory planning algorithm, Fig. 13(e) shows the position and velocity of EV during the whole driving process. And Fig. 13(f) shows the trajectories of all the vehicles, where the red curve is the trajectory of EV.

3.2. Testing scenario 2

The testing two is the scenario of changing lane to the right, which is shown as Fig. 12(b). The EV drives in the left-most lane, and can only follow the front vehicle in current lane in *Following* mode due to the right front vehicle in the right lane is moving at a slower speed and blocking its lane change space. When EV passes the right front vehicle, the behavior planning judges that lane change conditions are available and generates behavioral decision command CR, then the trajectory planning generates the optimal desired trajectory for changing lane to the right. As there is no other vehicles in front of the target lane, the EV enters the *VelocityKeeping* mode after reaching the limited speed. The results are shown in Fig. 14.

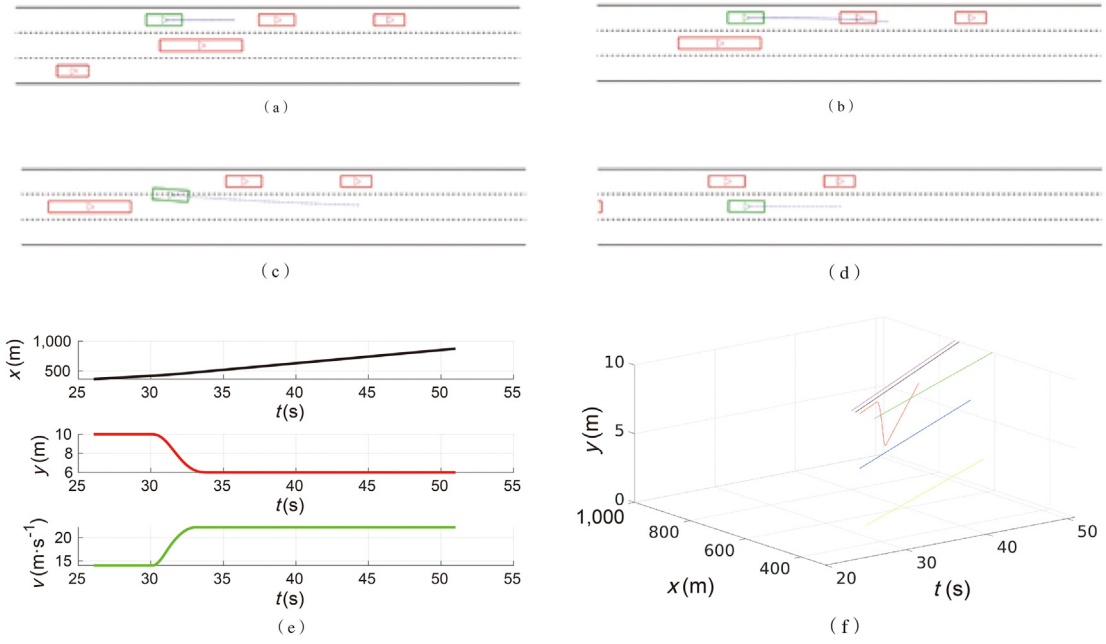


Fig. 14. Changing lane to the right.

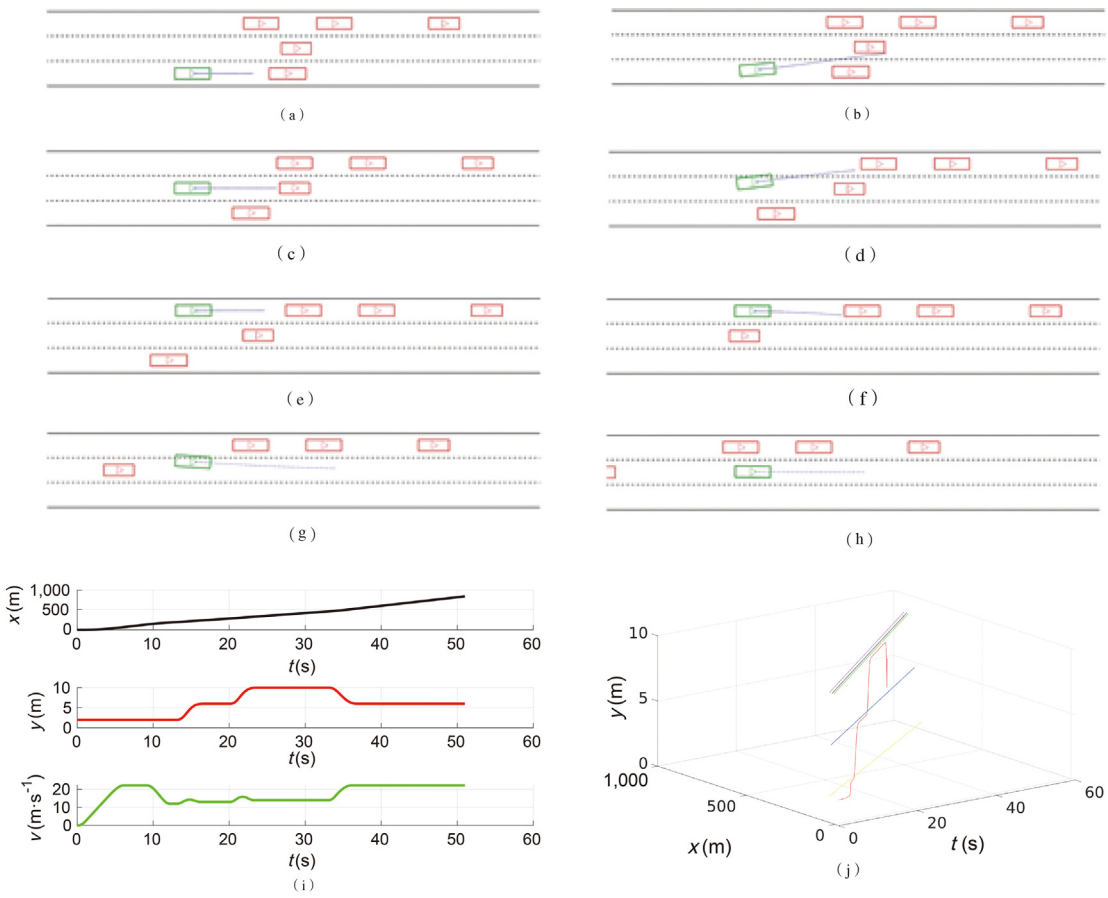


Fig. 15. Continuous lane change.

3.3. Testing scenario 3

The testing three is the continuous lane change scenario, which is shown as Fig. 12(c). The EV drives along the right-most lane, and the road speed limit is 80 km/h. After the vehicle reaches the limited speed, it enters the mode of *VelocityKeeping*. At this time, the EV founds that the speed of the vehicle in front is very slow, and the speed of the left front vehicle in the left lane is faster than the front vehicle. And the HMPF makes the behavior of CL and follows the front vehicle in *Following* mode. When it finds that the speed in the left lane is faster, the vehicle changes lane to the left again and follows the front vehicle. And when the right lane has the lane changing conditions, the EV completes the behavior of CR, and enters the *VelocityKeeping* after reaching the speed limit. The results are shown in Fig. 15, where Fig. 15(a) and (b) show the first changing lane to the left, Fig. 15(c) and (d) are the second, and Fig. 15(e)–(h) show the process of changing lane to the right.

4. Conclusions

This paper introduces a hybrid motion planning framework (HMPF) consist of learning-based behavior planning and optimization-based trajectory planning for autonomous vehicles in mixed traffic environments. This framework enables the motion planning system to have the learning ability while still having the security and reliability of traditional methods. Considering the interaction between the EV and HDVs, a DRL-based behavior planning is used to generate the behavioral command. Particularly, in order to more realistically simulate the bidirectional interaction between EV and HDVs, the IDM model calibrated with real vehicle driving data is employed to drive the HDVs, so that HDVs can make real-time interactive responses to EV's behavior. In addition, the optimization method based on the frenet coordinate can ensure that the trajectory planning generates a safe and comfortable desired trajectory in the complex dynamic urban environment. The simulation test results show that the proposed HMPF is effective and feasible.

In future work, we will consider individual differences of human drivers, such as driving styles and driving intentions, so as to improve the performance of the proposed method and generate driving behaviors more consistent with excellent human drivers.

Declaration of competing interest

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

Acknowledgement

This work was supported by the National Natural Science Foundation of China under Grants U19A2083.

References

[1] Katrakazas C, Quddus M, Chen W-H, Deka L. Real-time motion planning methods for autonomous on-road driving: state-of-the-art and future research directions. Transport Res C Emerg Technol 2015;60:416–42. https://doi.org/10.1016/j.trc.2015.09.011.

[2] Pendleton SD, Andersen H, Du X, Shen X, Meghjani M, Eng YH, et al. Perception, planning, control, and coordination for autonomous vehicles. Machines 2017;5(1):6. https://doi.org/10.3390/machines5010006.

[3] Paden B, Čáp M, Yong SZ, Yershov D, Frazzoli E. A survey of motion planning and control techniques for self-driving urban vehicles. IEEE Trans Intell Veh 2016;1(1):33–55. https://doi.org/10.1109/TIV.2016.2578706.

[4] Sun L, Peng C, Zhan W, Tomizuka M. A fast integrated planning and control framework for autonomous driving via imitation learning. http://arxiv.org/abs/1707.02515; 2017.

[5] González D, Pérez J, Milanés V, Nashashibi F. A Review of Motion Planning Techniques for Automated Vehicles17; 2016. p. 1135–45. https://doi.org/10.1109/TITS.2015.2498841 (4).
[6] Dolgov D, Thrun S, Montemerlo M, Diebel J. Path planning for autonomous vehicles in unknown semi-structured environments. Int J Robot Res 2010. https://doi.org/10.1177/0278364909359210.
[7] Ziegler J, Stiller C. Spatiotemporal state lattices for fast trajectory planning in dynamic on-road driving scenarios. In: 2009 IEEE/RSJ international conference on intelligent robots and systems, IROS 2009; 2009. https://doi.org/10.1109/IROS.2009.5354448.
[8] Bohren J, Foote T, Keller J, et al. Little ben: the ben franklin racing team's entry in the 2007 DARPA urban challenge. J Field Robot 2008. https://doi.org/10.1002/rob.20260.
[9] Kavraki LE, Švestka P, Latombe JC, Overmars MH. Probabilistic roadmaps for path planning in high-dimensional configuration spaces. IEEE Trans Robot Autom 1996. https://doi.org/10.1109/70.508439.
[10] Kavraki LE, Kolountzakis MN, Latombe JC. Analysis of probabilistic roadmaps for path planning. IEEE Trans Robot Autom 1998. https://doi.org/10.1109/70.660866.
[11] Karaman S, Frazzoli E. Sampling-based algorithms for optimal motion planning. Int J Robot Res 2011. https://doi.org/10.1177/0278364911406761.
[12] Berglund T, Brodnik A, Jonsson H, Staffanson M, Söderkvist I. Planning smooth and obstacle-avoiding B-spline paths for autonomous mining vehicles. IEEE Trans Autom Sci Eng 2010. https://doi.org/10.1109/TASE.2009.2015886.
[13] Brezak M, Petrovic I. Real-time approximation of clothoids with bounded error for path planning applications. IEEE Trans Robot 2014. https://doi.org/10.1109/TRO.2013.2283928.
[14] Biral F, Zendri F, Bertolazzi E, et al. A web based "Virtual Racing Car Championship" to teach vehicle dynamics and multidisciplinary design. In: ASME 2011 international mechanical engineering congress and exposition, IMECE 2011; 2011. https://doi.org/10.1115/imece2011-65245.
[15] Tavernini D, Massaro M, Velenis E, Katzourakis DI, Lot R. Minimum time cornering: the effect of road surface and car transmission layout. Veh Syst Dyn 2013. https://doi.org/10.1080/00423114.2013.813557.
[16] Kelly A, Nagy B. Reactive nonholonomic trajectory generation via parametric optimal control. Int J Robot Res 2003;22(7–8):583–601. https://doi.org/10.1177/02783649030227008.
[17] Kiran BR, Sobh I, Talpaert V, et al. Deep reinforcement learning for autonomous driving: a survey. IEEE Trans Intell Transport Syst 2022. https://doi.org/10.1109/TITS.2021.3054625.
[18] Zhang Y, Zhang J, Zhang J, Wang J, Lu K, Hong J. Integrating algorithmic sampling-based motion planning with learning in autonomous driving. ACM Trans Intell Syst Technol 2022;13(3). https://doi.org/10.1145/3469086.
[19] Lu H, Lu C, Yu Y, Xiong G, Gong J. Autonomous overtaking for intelligent vehicles considering social preference based on hierarchical reinforcement learning. Automot Innov 2022;5(2):195–208. https://doi.org/10.1007/s42154-022-00177-1.
[20] Wang H, Gao H, Yuan S, Zhao H, Wang K, Wang X, et al. Interpretable decision-making for autonomous vehicles at highway on-ramps with latent space reinforcement learning. IEEE Trans Veh Technol 2021;70(9). https://doi.org/10.1109/TVT.2021.3098321.
[21] Lu C, Wang H, Lv C, Gong J, Xi J, Cao D. Learning driver-specific behavior for overtaking: a combined learning framework. IEEE Trans Veh Technol 2018;67(8). https://doi.org/10.1109/TVT.2018.2820002.
[22] Lu C, Gong J, Lv C, Chen X, Cao D, Chen Y. A personalized behavior learning system for human-like longitudinal speed control of autonomous vehicles. Sensors 2019;19(17). https://doi.org/10.3390/s19173672.
[23] Fehér A, Aradi S, Hegedus F, Bécsi T, Gaspar P. Hybrid DDPG approach for vehicle motion planning. In: ICINCO 2019 - proceedings of the 16th international conference on informatics in control, Autom Robot. vol. 1; 2019. https://doi.org/10.5220/0007955504220429.
[24] Li L, Ota K, Dong M. Humanlike driving: empirical decision-making system for autonomous vehicles. IEEE Trans Veh Technol 2018;67(8). https://doi.org/10.1109/TVT.2018.2822762.
[25] Chen C, Seff A, Kornhauser A, Xiao J. DeepDriving: learning affordance for direct perception in autonomous driving. In: Proc IEEE int conf comput vis. 2015;2015 International Conference on Computer Vision, ICCV; 2015. p. 2722–30. https://doi.org/10.1109/ICCV.2015.312.
[26] Xiong X, Wang J, Zhang F, Li K. Combining deep reinforcement learning and safety based control for autonomous driving. 2016. p. 1–10. http://arxiv.org/abs/1612.00147.
[27] Moghadam M, Alizadeh A, Tekin E, Elkaim GH. A deep reinforcement learning approach for long-term short-term planning on frenet frame. In: IEEE international conference on automation science and engineering. Vol 2021-August. ; 2021. doi: 10.1109/CASE49439.2021.9551598.
[28] Bi H, Mao T, Wang Z, Deng Z. A deep learning-based framework for intersectional traffic simulation and editing. IEEE Trans Visual Comput Graph 2019:1–15. https://doi.org/10.1109/TVCG.2018.2889834.
[29] https://en.wikipedia.org/wiki/Intelligent_driver_model.
[30] Kesting A, Treiber M. Calibrating car-following models by using trajectory data methodological study. Transport Res Rec 2008;(2088). https://doi.org/10.3141/2088-16.
[31] Fletcher L, Teller S, Olson E, Moore D, Kline FR. The DARPA urban challenge: autonomous vehicles in city traffic. Springer Publishing Company, Incorporated; 2010.
[32] Sutton RS, Barto AG. Reinforcement learning: an introduction[M]. MIT press; 2018.
[33] Werling M, Kammel S, Ziegler J, Gröll L. Optimal trajectories for time-critical street scenarios using discretized terminal manifolds. Int J Robot Res 2012;31(3):346–59. https://doi.org/10.1177/0278364911423042.
[34] Werling M, Ziegler J, Kammel S, Thrun S. Optimal trajectory generation for dynamic street scenarios in a frenét frame. In: Proceedings - IEEE international conference on robotics and automation; 2010. https://doi.org/10.1109/ROBOT.2010.5509799.
[35] Zhang Y, Chen H, Waslander SL, et al. Toward a more complete, flexible, and safer speed planning for autonomous driving via convex optimization. Preprints 2018:1–29. https://doi.org/10.20944/preprints201805.0164.v2.
[36] Zhang Y, Chen H, Waslander SL, Gong J, Xiong G, Yang T. Hybrid trajectory planning for autonomous driving in highly constrained environments2; 2018. p. 1–18. https://doi.org/10.1109/ACCESS.2018.
[37] Dosovitskiy A, Ros G, Codevilla F, Lopez A, Koltun V. CARLA: an open urban driving simulator. CoRL 2017:1–16. http://arxiv.org/abs/1711.03938.