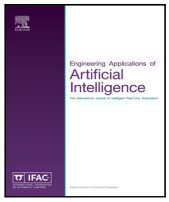
Engineering Applications of Artificial Intelligence 123 (2023) 106291

Contents lists available at ScienceDirect

Engineering Applications of Artificial Intelligence

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



Trajectory planning based on spatio-temporal reachable set considering dynamic probabilistic risk

Xinkang Zhang[1], Bo Yang[2], Xiaofei Pei *, Songxin Lu[3]

Hubei Key Laboratory of Advanced Technology of Automotive Components, Wuhan University of Technology, Wuhan 430070, China
Hubei Collaborative Innovation Center of Automotive Components Technology, Wuhan University of Technology, Wuhan 430070, China

ARTICLE INFO

Keywords:
Intelligent vehicle
Spatio-temporal trajectory planning
Probabilistic risk
Reachable sets

ABSTRACT

Trajectory planning in complex traffic situations has always been a challenging task for intelligent vehicles. Comparing to the decoupling method, the spatio-temporal trajectory planning method owns more flexibility and reasonability due to the combination of lateral and longitudinal motion. However, it still has some shortcomings, such as unreasonable risk assessment, high computational complexity and heavy dependence on other models for generating target points. Therefore, a novel spatio-temporal based trajectory planning framework considering probability risk is proposed in this paper. Firstly, a GNN-LSTM based on trajectory prediction algorithm is presented in terms of risk analysis, and particularly the predicted trajectories and the vehicle dynamic model are combined for risk assessment. Secondly, a rough-fine hierarchical planning framework based on reachable set and dynamic programming is proposed. In this framework, the reachable set is taken as spatio-temporal node to reduce the computational costs and dynamic programming can help the algorithm to eliminate the dependence of other models in evaluating target points. Finally, a traffic scenario with random interactive obstacles is built and tested on a HIL platform. The experimental results show that compared with other typical algorithms, the average driving efficiency, driving risk behavior and driving comfort of the vehicle are significantly improved.

1. Introduction

Trajectory planning in complex traffic scenarios has been one of hot issues of intelligent vehicle in recent years. It needs to combine traffic rules, perceptual information and obstacle information to generate a collision-free trajectory in all road scenarios, and in the meantime offers the passengers sufficient comfort (Nyberg et al., 2021). Therefore, it requires a performance trade-off between safety, driving efficiency, optimality and comfort.

Similar to mobile robots, traditional trajectory planning of autonomous vehicles divides trajectory planning into path planning and speed planning and then carries out step-by-step planning (Hu et al., 2018). But in fact, the impact of the first step planning results on the subsequent step planning cannot be evaluated in step-wise planning under dynamic scenarios (Xin et al., 2021). Because of this limitation, the planned trajectory is usually not reasonable and smart enough in high-speed scene. Therefore, Ajanovic et al. (2018) proposed a spatio-temporal trajectory planning method. By introducing the time axis, the planning space is merged from two planes into an overall three-dimensional space to better handle dynamic obstacles and the integrity of path and speed planning at the same time.

1.1. Spatio-temporal planning related work

At present, most planning methods based on spatio-temporal maps are implemented in the two-layer architecture (Katrakazas et al., 2015): rough search for obstacles and traffic rules in spatio-temporal maps to obtain rough results. These results are usually composed of discrete spatio-temporal points or driving corridors. The second step is to calculate the final planning trajectory curve in terms of driving comfortness under the rough search results (Sharma et al., 2021).

According to current research on spatio-temporal planning, rough planning is a step that directly determines the efficiency and safety of vehicle driving corridor. Currently, rough planning methods are mostly implemented based on a set of maps with discrete time steps, while the spatio-temporal driving corridor based on machine learning needs further study. The connection relationship between maps in adjacent time steps is derived from the designed equation based on the

* Correspondence to: Hubei Research Center for New Energy & Intelligent Connected Vehicle, Wuhan University of Technology, Wuhan, 430070, China.
E-mail addresses: whut_zxk@126.com (X. Zhang), yangbo92@126.com (B. Yang), peixiaofei7@whut.edu.cn (X. Pei), Lsx1324764580@whut.edu.cn (S. Lu).
[1] Xinkang Zhang is with the Department of Automotive Engineering, Wuhan University of Technology, Wuhan, 430070, China.
[2] Bo Yang is with the Hubei Key Laboratory of Advanced Technology for Automotive Components, Wuhan University of Technology, Wuhan, 430070 China.
[3] Songxin Lu is with the Department of Automotive Engineering, WuhanUniversity of Technology, Wuhan, 430070, China.

https://doi.org/10.1016/j.engappai.2023.106291
Received 30 August 2022; Received in revised form 30 March 2023; Accepted 4 April 2023
Available online 17 April 2023

vehicle dynamics model. Based on the different descriptions methods for spatio-temporal map, the rough planning methods can be divided into *Discretization-based method* and *Set-based method.*

(1) *Discretization-based method:* In this method, the spatio-temporal map is constructed by sampling discrete state points in each time step and generating the connection of state points with vehicle model. Based on this abstract topological structure, search-based method is one of the most widely used algorithms. The target end point is obtained by preset rules or decision algorithm, and the safe trajectory is solved by A* (Zhang et al., 2022) or hybrid A* (Xin et al., 2021). However, this algorithm needs to require prior knowledge or decision model to determine a search target point, which is critical in the quality of planning trajectory. In addition, due to the limitations of discrete action space, actions usually only consider discrete values in a limited range, thus not all feasible solutions can be covered. Some other researchers used sampling-based methods to solve driving corridor, which are random or deterministic sampling in spatio-temporal maps (Rocamora and Pereira, 2022), represented by RRT* (Karaman and Frazzoli, 2011) and Lattice Planner (Rocamora and Pereira, 2022; Saini et al., 2021). But the asymptotic convergence of RRT* makes it difficult to guarantee both the optimality of the path and the real-time performance of the algorithm. Lattice planner discretizes spatio-temporal space into uniform lattices, and describes the transfer relationship between lattices through designed curves (McNaughton et al., 2011). However it becomes difficult to achieve lower computational complexity in the case of dense obstacles (Qian et al., 2022).

(2) *Set-based method:* Unlike the method of describing spatio-temporal space as discrete state points, the set based method only focuses on the states that can be reached by the ego vehicle, and obtains the rough planning solution through segmentation and screening of the state sets. The set-based approach is utilized to first to calculate all accessible state sets of the vehicle in time and space based on the vehicle motion model, then to evaluate the driving corridor by searching for the optimal convex set at the last moment and finally determine the backward reachable set (Sontges and Althoff, 2018). This approach avoids searching in the huge discrete grid to effectively reduce the computational complexity. It also ensures that the optimization boundary can be provided such that subsequent fine optimization is convex, thus reducing the processing loads in the middle stage of the two-layer programming. However, the disadvantage of Sontges' method include that only the last-minute gains are considered when calculating the driving corridor through the reachable set, only feasibility is considered for the area in the process, and risk and benefit are both not considered. The risk level is represented by the size of the aggregate area, which is not reliable and reasonable in the actual planning process.

Fine planning is responsible for further solving the smooth trajectory on the basis of rough planning. At present, numerical optimization has become the most extensive fine planning method. Based on convex constraints in rough programming, it solves a series of spatial trajectory points. Xin et al. (2021) uses model predictive control (MPC) and considers the dynamic bicycle model to solve the fine trajectory. The optimization objectives mainly include optimality and vehicle dynamics stability. However, its real-time performance is poor. Manzinger et al. (2021) simplifies MPC into two parts of quadratic programming (QP) for optimization, and represents the optimization objective and vehicle dynamic model as several kinematic factors, mainly including longitudinal and lateral accelerations, optimality and traffic rules. Under a given path, the velocity is planned based on velocity, acceleration and jerk, and the degenerate algorithm is implemented in case of the failure of the solution (Artunedo et al., 2022).

To summarize, it can be seen that the current research focuses on how to design an framework to reasonably and quickly calculate the spatio-temporal trajectory. However, there lacks extensive research on risk assessment and the predicted trajectories of obstacles and risk assessment are usually ideal. Therefore, how to reasonably carry out risk assessment in spatio-temporal trajectory planning algorithm is valuable.

1.2. Risk-aware related work

In spatio-temporal planning algorithm, the risk assessment of obstacles cannot be limited to the current time step, but it needs to evaluate the risk of the trajectory over a continuous time period. Therefore, the risk characterization can be divided into two parts: obstacle trajectory prediction and obstacle risk assessment at each time step.

1.2.1. Trajectory prediction methods

Currently, common trajectory prediction methods can be divided into two categories: physical mechanism-based model and behavior-based model.

The physical model only considers the current state of motion of the vehicle, and the related vehicle model is used to forecast the future trajectory. This method is usually implemented based on kinematic model, such as CTRA model (Kim et al., 2019), which regards the future movement of vehicle as the movement with fixed acceleration and fixed yaw rate. Consequently, it limits the prediction within short times.

The behavior based model can predict the future trajectory of vehicles by considering the relevant results by behavior perception. Altche et al. (2017) first utilizes neural networks to evaluate future prediction trajectories based on historical information, but this one-step method has poor reliability and strong dependence on data sets. Then, a hierarchical prediction algorithm is proposed to first consider the driver's intention to predict the trajectory. The main methods include Bayesian network and LSTM network. However, this method only considers a single obstacle. In recent years, many researches have investigated multi-vehicle interaction using such methods, such as various complex layered network frameworks based on LSTM. This incorporation of multi-vehicle interaction and vehicle behavior intention has proved to be more suitable for the long-term prediction in complex traffic environment.

1.2.2. Risk assessment methods

Risk assessment mainly includes online sampling based method, learning based method and approximate model based method. Sampling based methods are implemented according to Monte Carlo simulation and variation inference (Li et al., 2022), which use a large number of samples to approximate inference. Because this method requires online sampling, it usually needs a large amount of calculation and therefore the real-time performance of the algorithm cannot be guaranteed.

Learning based algorithms are mainly used to solve a variety of uncertainty conditions in an end-to-end manner. The representative methods are deep Bayesian networks, reinforcement learning, and imitation learning (Liu et al., 2021). For these methods, there is no need of complex models, but the disadvantages lie in that the generalization ability is poor, the interpretation is not strong, and the effect cannot be guaranteed in a changing environment.

The model-based approach is adopted to design a risk model based on the distribution of obstacles. Generally, the environmental risk model is obtained through the bounded uniform distribution or Gaussian distribution density function, combined with the momentum based collision severity assessment (Hruschka et al., 2019; Khaitan et al., 2021). The model-based approach does not require a large number of online samples. Compared with the sampling method, the computational complexity can be greatly reduced, and the structure is clearer and more interpretable. The disadvantage is that the uncertainty risk model needs to be carefully designed, and different models need to be designed for different uncertainty conditions, which is very difficult for the designers.

1.3. Contributions

To solve the above issues, we propose a layered trajectory planning method which combines probabilistic risk assessment and reachable sets. Our main contributions include:

1. We propose an obstacle trajectory prediction algorithm based on GNN-LSTM and a probability risk assessment method based on dynamic model. The prediction accuracy is improved by considering the multi-vehicle interaction. Based on the dynamic characteristics of the predicted trajectory, the trend of the position distribution uncertainty of obstacles in the space–time domain is evaluated.

2. To deal with the shortcomings of current reachable sets methods, a risk-sensitive set partition method is proposed. By this method, the risks and benefits of the process can be comprehensively considered to avoid falling into local optimum, and there is no need for other algorithms to provide search target points.

3. We solve the optimal trajectory by performing twice quadratic programming to measure optimality and comfort. At the same time, a safety checking model is designed at the fine planning level to ensure the safety of the trajectory.

1.4. Organization

The rest of this paper is presented in the following. In Section 2, the relevant problem is formulated and the relating background is introduced. Sections 3–5 describe the overall structure of the algorithm, the probability risk assessment method based on LSTM and dynamic model, and the spatio-temporal trajectory planning algorithm of reachable sets based on probability risk improvement, respectively. Section 6 discusses the experimental verification of the algorithm and the corresponding results. Finally, the conclusions are given in Section 7.

2. Problem formulation and preliminaries

This section mainly introduces the prior knowledge and mathematical modeling of the problem to be solved. Since the scenario in this paper focuses on a steady traffic flow under structured road, we can simplified the model characteristics of vehicle dynamic in the planning.

2.1. Problem statement

Our objective is to calculate the planning trajectory in the Frenet coordinate for a given set of obstacle information and road structure, which should both consider safety, driving efficiency, traffic rules and driving efficiency. The planning trajectory is described as $\{s_0, p_0, t_0; \ldots; s_T, p_T, T\}$. From the projection curve of this trajectory, the planned path curve and the planned speed curve can be obtained.

Vehicle dynamics system is often complex and non-linear. In this paper, the states of the vehicle in the spatio-temporal coordinate system can be briefly represented as $S(s, p, vx, vy, t)$, where s and p respectively denotes the distance and lateral distance of the vehicle in the Frenet coordinate system, and vx and vy represent the longitudinal and lateral speed of the vehicle in the Frenet coordinate system, respectively. The time interval from the current time t_0 to the prediction time T is discretized into discrete time points and the time interval between adjacent time points is denoted as Δt.

2.2. Reachable set

Reachable set refers to the set of all the states that a vehicle can reach in the future in its current state. Generally, only longitudinal and lateral positions and speeds are considered for the vehicle state, and its schematic diagram is shown in Fig. 1.

The reachable set is represented as an area in the X-Y plane, such as the gray rectangle in the upper part of Fig. 1. In fact, the reachability set determines not only the spatial boundaries, but also the location-dependent speed boundaries, such as the curvilinear boundaries of the x-vx and y-vy planes in the lower half part of Fig. 1.

In the Frenet coordinate system, the vehicle state transition can be formulated corresponding to the basic dynamic model:

$$\begin{cases} s = s_0 + \int_{t0}^{T} f_s(s(t), vx(t), ax(t))dt \\ p = p_0 + \int_{t0}^{T} f_p(p(t), vy(t), ay(t))dt \\ vx = vx_0 + \int_{t0}^{T} f_{vx}(vx(t), ax(t))dt \\ vy = vy_0 + \int_{t0}^{T} f_{vy}(vy(t), ay(t))dt \end{cases} \tag{1}$$

Then the reachable set $\boldsymbol{Reach}(T; S_0)$ with $S_0(s_0, p_0, vx_0, vy_0, t_0)$ denoted as the initial state can be defined as: the set of all the states that a vehicle with S_0 can reach in the t-period, which can be expressed as:

$$\begin{aligned} \boldsymbol{Reach}\left(T; S_0\right) = \{ & S\left(T, u, S_0\right) | \exists u \in \mathcal{U}, \\ & S\left(t, u, S_0\right) \notin \mathcal{F}(s, p, t)\, int \in [t_0, t_0 + T] \} \end{aligned} \tag{2}$$

where u is the input of the system, here denoted as (ax, ay), $\mathcal{U}$ is the range of the lateral and longitudinal accelerations, assuming that the two accelerations will not affect each other and $\mathcal{F}(s, p, t)$ is an forbidden region. When a state $S_1(s_1, p_1, vx_1, vy_1, t_1)$ does not belong to set $\mathcal{F}$ and there has a set of inputs u such that all states in the process from S_0 to S_1 are not in set $\mathcal{F}$, this state will be included by $\boldsymbol{Reach}\ (T; S_0)$.

3. Framework

The overall architecture of the proposed spatio-temporal trajectory planning algorithm is illustrated in Fig. 2. Upstream of risk assessment has a sensor module, which is responsible for sensing and recording the historical trajectory of obstacles based on the lidar, radar and camera. At the same time, traffic rules information is provided based on high-precision map. Our work mainly includes obstacle prediction and risk assessment, as well as hierarchical trajectory planning. Risk distributions in the environment can be evaluated from upstream environment information and obstacle trajectories prediction. Next, the trajectory planning is divided into rough planning and fine planning. Rough planning computes the reachable sets of lateral and longitudinal dimensions, divides the reachable subset based on risk level, and then obtains driving corridor by dynamic programming. In the fine planning process, the lateral and longitudinal quadratic programming is used to solve the smooth trajectory based on the driving corridor.

The fine planning has a fixed re-planning cycle which is 10 ms for updating the smooth trajectory in real time. Rough planning has a long predicated distance. In order to avoid unnecessary repeated planning, the rough planning does not have a fixed re-planning frequency, but relies on a separate safety verification module to trigger.

The safety check module is a high-frequency cycling module that checks the remaining available length of the driving corridor at each step, and also trajectory safety for a short predication time. The trigger module is used to activate rough planning to recalculate the driving corridor. The trigger process of replanning is shown in Algorithm 1. At each iteration, the availability and safety of planned trajectory are checked. Firstly, the position of vehicle current status $\boldsymbol{S_0}$ in the generated corridor $\boldsymbol{D_c}$ is located, and the remaining available distance of the driving corridor is obtained. If it is less than 40% of the total length of the corridor, it is judged that the driving corridor needs to be updated. Safety checks are used to determine if a driving corridor is at risk by performing collision detection on track $\mathfrak{X}'$ generated by the previous step and obstacle short-term trajectory point $(\boldsymbol{ot_0} - \boldsymbol{ot_T})$. The short-term trajectory of obstacles required for short-term safety verification of the module is obtained by CTRA model (Li et al., 2022).

The motion control can be divided into lateral control and longitudinal control. The steering angle of vehicle is obtained by lateral control which contains a feedforward controller and a feedback controller. The feedforward controller is used to track the road curvature and the feedback controller correcting the error of lateral displacement and heading angle. The specific algorithm can be found in the literatures (Szepe and Assal, 2012; Liu et al., 2019).

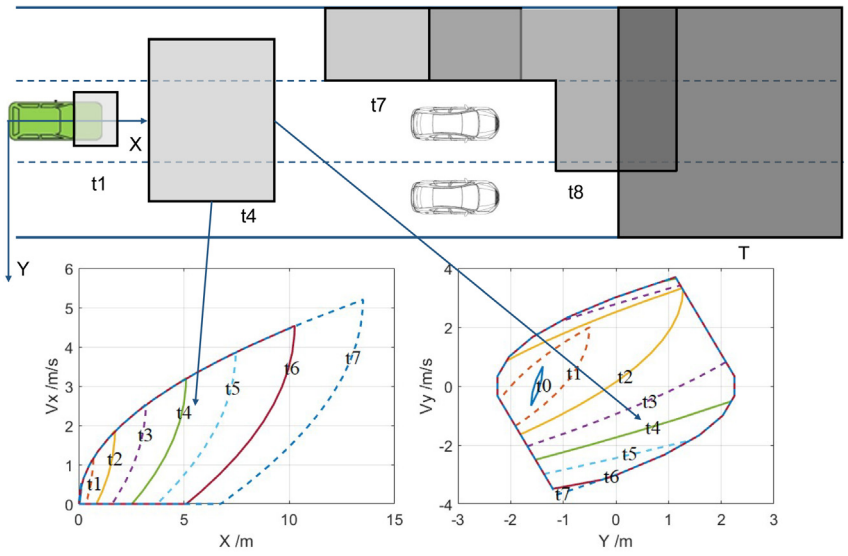


Fig. 1. Reachable set.

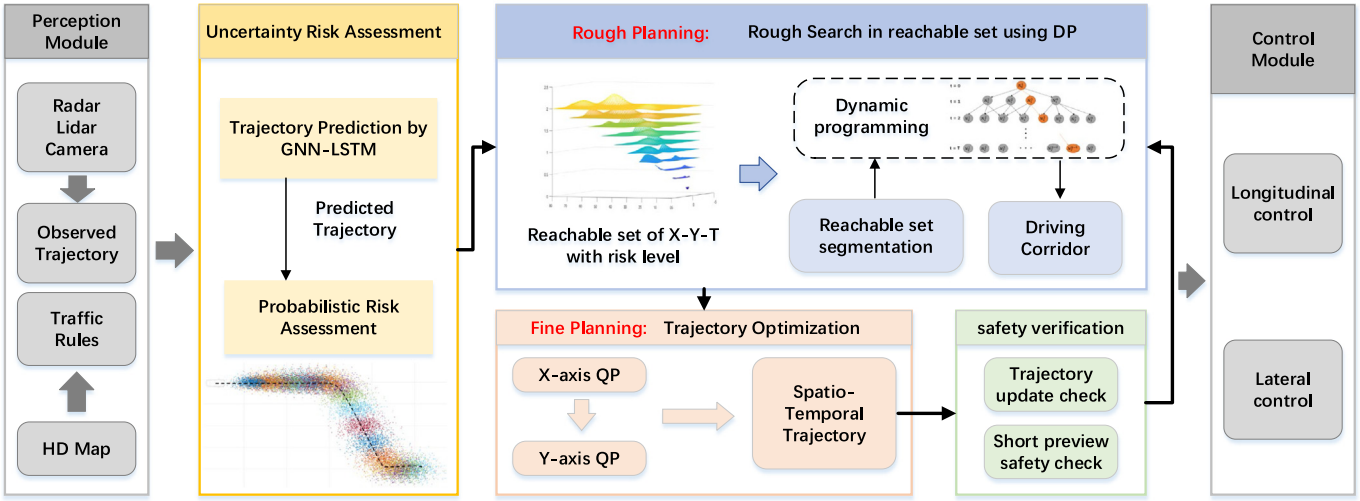


Fig. 2. Framework of the proposed spatio-temporal trajectory planning.

4. probabilistic risk assessment

The risk assessment process in this paper is divided into three steps, as shown in Fig. 3. First, based on the historical trajectories of the surrounding obstacles obtained by the perception module, the prediction trajectories of the obstacles are obtained. Then the prediction trajectory of each obstacle is represented in probability and the deterministic trajectory is depicted as a probability distribution at each time step based on vehicle dynamic model. After the location distribution of each obstacle is obtained, the collision probability is obtained by sampling within the reachable area to characterize the risk level.

4.1. Trajectory prediction

In this section, we propose a GNN-LSTM network to learn the interaction between the main vehicle and the surrounding vehicles in time and space dimensions, and use the attention mechanism to more reasonably consider the impact of the surrounding vehicles to predict their trajectories (Altche et al., 2017; Kalatian and Farooq, 2022).

In this paper, the predicate network considers the trajectory information of the main vehicle and the six surrounding vehicles, and the GNN-LSTM network framework is shown in Fig. 4, which consists of two parts. The encoder is responsible for understanding and memorizing the characteristics of historical trajectories from the input data, and the decoder extracts the understanding results from the sequential input based on the output of the encoder and generates the future trajectory. In the encoding stage, the graph structure is used to reflect the interactive characteristics of space dimension and time dimension between the vehicles, and the attention mechanism is used to filter the concerned environmental information. In the decoding stage, LSTM network is used to analyze the output of graph neural network and evaluate the predicate results. Finally, the predicate values are compared with the actual values to evaluate the expected values and variances of vehicle position prediction errors.

In order to better consider the interaction between the vehicles, a graph-based framework is used in the encoder. The predicate vehicle represents the node in graph structure. Relative information between two vehicles in spatial dimension and that of the same vehicle in time dimension are represented as edges in graph structure. LSTM network is applied to the nodes and edges. Graph structure G is defined as:

$$G = (N_{vehicle}, E_{spatio}.E_{temporal}) \tag{3}$$

where node $N_{vehicle}$ represents the characteristics f of the vehicle, edge E_{spatio} denotes the interaction characteristics of each vehicle in the

Algorithm 1: Trajectory Replanning Trigger

Input:

Vehicle initial states $\boldsymbol{S_0}$,

Obstacle Predicted trajectory $(\boldsymbol{ot_0 - ot_T})$,

Replanning Flag $\boldsymbol{\mathcal{F}s}$, Driving Corridor $\boldsymbol{D_c}$,

Last time Planned Trajectory $\boldsymbol{\mathcal{X}'}$

Output: Trajectory $\boldsymbol{\mathcal{X}}: (\boldsymbol{x_0, y_0, t_0})\boldsymbol{to}(\boldsymbol{x_T, y_T, t_T})$

1. Initialize $\boldsymbol{\mathcal{X}}$ **wih** $\emptyset$
2. **If** $\mathit{TrajectoryUpdateCheck}(\boldsymbol{D_c}, \boldsymbol{S_0}) <= 40\%$
3. $\boldsymbol{\mathcal{F}s} = \boldsymbol{true}$
4. **Elseif** $\mathit{SafyCheck}(\boldsymbol{\mathcal{X}'}, \boldsymbol{ot_0 - ot_T}) == \boldsymbol{true}$
5. $\boldsymbol{\mathcal{F}s} = \boldsymbol{true}$
6. **Endif**
7. **If** $\boldsymbol{\mathcal{F}s == true}$
8. $\boldsymbol{D_c} = \mathit{Roughplan}(\boldsymbol{S_0}, \boldsymbol{ot_0 - ot_T})$
9. **Endif**
10. $\boldsymbol{\mathcal{X}} = \mathit{TrajectoryOptimization}(\boldsymbol{D_c}, \boldsymbol{S_0})$

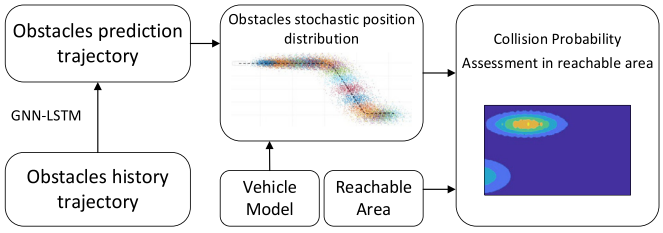


Fig. 3. Risk assessment process.

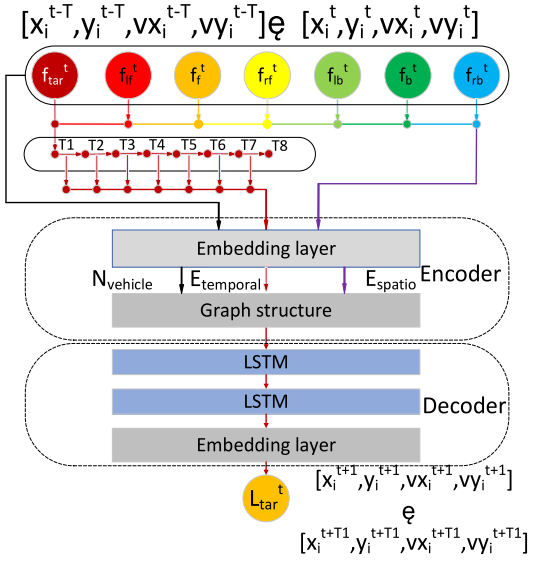


Fig. 4. GNN-LSTM prediction structure.

space dimension and edge $E_{temporal}$ illustrates the interaction characteristics of the vehicle in the time dimension. At moment t, the interaction characteristics in the spatial dimension between vehicle I and vehicle J can be represented by:

$$E_{spatio}^{t,ij} = \left(N_{vehicle-i}^{t}, N_{vehicle-j}^{t}\right) = f_{ij}^{t} = \left(x_{ij}^{t}, y_{ij}^{t}, vx_{ij}^{t}, vy_{ij}^{t}\right) \tag{4}$$

where wx_{ij}^{t}, y_{ij}^{t}, vx_{ij}^{t} and vy_{ij}^{t} represent the relative longitudinal and lateral positions, relative longitudinal and lateral speeds between vehicle I and vehicle J, respectively. At moment t, the interaction characteristics of vehicle I in the spatial dimension can be represented by:

$$E_{temporal}^{t,ii} = \left(N_{vehicle-i}^{t-1}, N_{vehicle-i}^{t}\right) = f_{ii}^{t} \tag{5}$$

These interactive features are aggregated by LSTM networks. For node $N_{vehicle}$, edge E_{spatio} and edge $E_{temporal}$, each LSTM layer is assigned to predict, and each part of the corresponding LSTM shares the same network parameters. In addition, Layer Normalization (LN) is used before entering into each LSTM layer, which prevents the occurrence of over-fitting to some extent. To quantify the significance of these interaction features, the attention module is used in the graph structure. Also, soft attention mechanism is used to assign different weights to spatial features to filter the interactive information that vehicles concern.

For training, the data set is first generated by sampling the trajectories of the ego vehicle and the surrounding vehicles in Prescan. Six surrounding vehicles are left front, front, right front, left rear, rear and right rear. Each vehicle collects four features: the longitudinal, lateral positions, the longitudinal and lateral speeds. If there is no vehicle in an area, the corresponding feature is set to 0. The sample range of vehicle trajectories is 15 s (8 s for history and 7 s for prediction). Secondly, these features are processed to acquire the vehicle and interaction information, and a training set containing 77 638 samples and a test set containing 19 410 samples are randomly selected. Finally, these information is input to the corresponding GNN-LSTM network model.

A series of gradient parameters of the same structure neural network model are set up, and their training results are compared to select the

Table 1
Error comparison of different parameters.

	Prediction horizon(s)	1	2	3	4	5	6
FDE(m)	(32,32)	0.501	1.260	2.033	2.899	3.830	4.708
	(64,64)	0.449	1.132	1.819	2.580	3.371	4.113
	(128,64)	0.428	1.075	1.750	2.495	3.263	3.998
	(128,128)	0.379	0.943	1.520	2.145	2.822	3.525
	(64,128)	0.392	0.975	1.547	2.167	2.826	3.514
ADE(m)	(32,32)	0.220	0.923	1.678	2.499	3.412	4.321
	(64,64)	0.197	0.827	1.505	2.231	3.016	3.788
	(128,64)	0.189	0.786	1.441	2.154	2.919	3.672
	(128,128)	0.168	0.692	1.257	1.859	2.515	3.209
	(64,128)	0.173	0.717	1.287	1.883	2.528	3.203

parameter configuration with the minimum prediction error. Finally, it is found that when 64 neurons are used in the embedded layer and 128 neurons are used in the LSTM layer, the performance is optimal. The comparison results are shown in Table 1. Where, FDE is the position error of the last time step, and ADE is the average position error. The LSTM layer uses softsign as the activation function, where the batch size is 64 and the learning rate is set to 0.0001. The model is trained 400 rounds through Adam optimizer, and the global norm of the gradient is clipped to 1 to ensure stable training.

After the training is completed, the GNN-LSTM model outputs predictions of the vehicle's longitudinal and lateral speeds, as well as the vehicle's lateral and longitudinal positions. Heading angles can be derived from the speed and initial positions. However, the deterministic prediction trajectory does not represent the risk of the obstacle over a period of time, and the uncertainty of location needs to be further evaluated.

Finally we compare our proposed method with CTRA, LSTM, GNN-LSTM (with or without attention), and the result in Table 2 shows that the accuracy of our algorithm is 75.59% higher than that of classic LSTM.

4.2. Model-based stochastic position distribution

Because of the uncertainty of the future movement of the obstacle, the description of the collision risk through the deterministic outline of the obstacle cannot meet the requirements of planning in dynamic scenes (Hruschka et al., 2019; Khaitan et al., 2021). Instead, it is more reasonable to use probability risk assessment (Candela et al., 2021; Guo et al., 2020).

We propose a method for calculating obstacle position distribution based on predicted trajectory and dynamic model. The predicted data can be represented as $(X_t, Y_t, Vx_t, Vy_t, \theta_t)$, where (Vx_t, Vy_t) refers to the longitudinal and lateral speeds in the road coordinate system. First, we need to convert the reference line of the Frenet coordinate system from the road centerline to the predicted trajectory. The converted information can be represented as $(X'_t, Y'_t, Vx'_t, Vy'_t, \theta'_t, ax'_t, Yawrate'_t)$ shown in Fig. 5, where $(ax'_t, Yawrate'_t)$ are derived from the difference between velocity information. Theoretically, the horizontal and vertical directions of this coordinate system will always coincide with those of the vehicle coordinate system, so the decomposed longitudinal and lateral movements in this coordinate system can be considered to be independent from each other (Ge et al., 2019). Dynamic modeling and probability distribution estimation can be performed separately.

In longitudinal motion, the acceleration is considered constant in a single time step, with t_0 as the starting point of the prediction and Δt as the discrete time interval. X'_t and Vx'_t can be expressed as:

$$X'_t = X'_{t-1} + Vx'_{t-1}\Delta t + 0.5ax'_{t-1}\Delta t^2 \tag{6}$$

$$Vx'_t = Vx'_{t-1} + ax'_{t-1}\Delta t \tag{7}$$

In lateral motion, the lateral velocity Vy'_t after coordinate system conversion is often affected by slip angle and errors in the heading angle. Since the slip angle generally has much less influence than the heading angle error, only the effect of the latter one is considered. the vehicle is assumed to be moving at a fixed heading angle in a single time step. Then lateral distance Y'_{t+1} and lateral velocity Vy'_t can be expressed as:

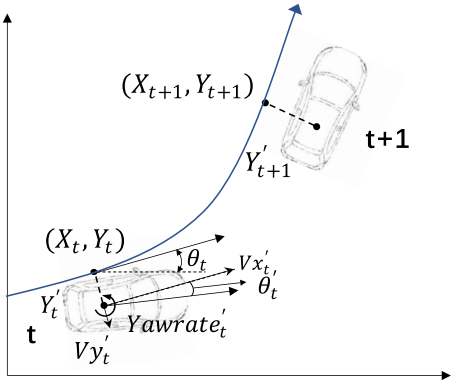


Fig. 5. Frenet coordinate system conversion.

$$Y'_{t+1} = Y'_t + Vy'_t\Delta t \tag{8}$$

$$Vy'_t = Vx'_{t-1}\sin\left(\theta'_{t-1} + Yawrate'_{t-1}\Delta t\right) \approx Vx'_{t-1}(\theta'_{t-1} + Yawrate'_{t-1}\Delta t) \tag{9}$$

As $\theta'_{t-1} + Yawrate'_{t-1}\Delta t$ is usually small, we can set $\sin(\theta) \approx \theta$.

By analyzing the uncertainty of the predicted position of the obstacle, it is known that the uncertainty of the longitudinal position is caused by the uncertainty of the longitudinal initial position, the longitudinal initial velocity and the prediction of the longitudinal acceleration. The lateral position uncertainty is caused by the uncertainty of lateral initial position, the lateral initial velocity and the predicted yaw rate. Therefore, the longitudinal and lateral position distribution can be expressed as:

$$\begin{cases} \mathbf{X}'_t = \mathbf{X}'_{t-1} + \mathbf{Vx}'_{t-1}\Delta t + 0.5\mathbf{ax}'_{t-1}\Delta t^2 & t > 1 \\ \mathbf{X}'_t = \mathbf{X}'_0 + \mathbf{Vx}'_0\Delta t + 0.5\mathbf{ax}'_0\Delta t^2 & t = 1 \end{cases} \tag{10}$$

$$\begin{cases} \mathbf{Y}'_t = \mathbf{Y}'_{t-1} + Vx'_{t-1}(\theta'_{t-1} + \mathbf{Yawrate}'_{t-1}\Delta t)\Delta t & t > 1 \\ \mathbf{Y}'_t = \mathbf{Y}'_0 + Vx'_0(\theta'_0 + \mathbf{Yawrate}'_0\Delta t)\Delta t & t = 1 \end{cases} \tag{11}$$

We uses bold fonts to represent the distribution of random variables. The uncertainty of the longitudinal velocity is not considered in the lateral motion. Assume that initial position $\mathbf{X}'_0, \mathbf{Y}'_0$, initial speed $\mathbf{Vx}'_0$, initial heading angle $\boldsymbol{\theta}'_0$, predicted acceleration $\mathbf{ax}'_t$ and predicted yaw rate $\mathbf{Yawrate}'_t$ obey different particular Gaussian distributions. The random position distribution at each time is derived from previous position distribution and velocity distribution by recursion from the formula (10) and (11). The distribution of $\mathbf{Vx}'_t$ and $\boldsymbol{\theta}'_t$ at each time is also evaluated from the distribution at the previous time by:

$$\mathbf{Vx}'_t = \mathbf{Vx}'_{t-1} + \boldsymbol{ax}'_{t-1}\Delta t \tag{12}$$

$$\boldsymbol{\theta}'_t = \boldsymbol{\theta}'_{t-1} + \mathbf{Yawrate}'_{t-1}\Delta t \tag{13}$$

The uncertainty of all initial states is caused by perception errors, so the variance of the Gaussian distribution of initial positions and speeds is fixed. The uncertainty of the future position is related to the future behavior of the vehicle. It is known that the greater the predicted acceleration and yaw rate are, the greater the uncertainty of future behavior will be Candela et al. (2021). Based on this principle, all of the Gaussian distribution are shown in Table 3:

Due to the additivity of Gaussian distribution, it can be concluded that the longitudinal and lateral position distributions at time t obey the Gaussian distribution with mean values of X'_t and Y'_t. The variance

Table 2
Error comparison of prediction algorithms.

	Prediction horizon(s)	1	2	3	4	5	6
FDE(m)	CTRA	0.326	1.202	2.594	4.476	6.787	9.527
	LSTM	0.457	1.265	2.266	3.490	4.846	6.198
	GNN-LSTM(without attention)	0.388	0.981	1.588	2.241	2.943	3.649
	GNN-LSTM	0.392	0.975	1.547	2.167	2.826	3.514
ADE(m)	CTRA	0.132	0.765	1.926	3.590	5.714	8.253
	LSTM	0.198	0.890	1.796	2.923	4.230	5.599
	GNN-LSTM(without attention)	0.171	0.716	1.311	1.942	2.625	3.334
	GNN-LSTM	0.173	0.717	1.287	1.883	2.528	3.203

Table 3
Random variable distribution.

Random variable	Distribution
Initial longitudinal position	$\mathbf{X}_0' \sim \mathcal{N}(X_0', \sigma_p^2)$
Initial lateral position	$\mathbf{Y}_0' \sim \mathcal{N}(Y_0', \sigma_p^2)$
Initial longitudinal velocity	$\mathbf{Vx}_0' \sim \mathcal{N}(Vx_0', \sigma_v^2)$
Initial heading angle	$\boldsymbol{\theta}_0' \sim \mathcal{N}(\theta_0', \sigma_\theta^2)$
Predicted acceleration	$\mathbf{ax}_t' \sim \mathcal{N}(ax_t', (ax_t')^2\sigma_{ax}^2)$
Predicted yaw rate	$\mathbf{Yawrate}_t' \sim \mathcal{N}(Yawrate_t', (Yawrate_t')^2\sigma_{yawrate}^2)$

$Var\left(X_n'\right), Var\left(Y_n'\right)$ of the two distributions at the nth time is expressed as:

$$Var\left(X_n'\right) = \sigma_p^2 + \sum_{i=1}^{n}(\sigma_{ax}^2 \frac{(ax_{i\Delta t}')^2 \Delta t^4}{4\lambda^i} + \frac{\Delta t^2(\sigma_v^2 + \sum_{j=1}^{i}(ax_{j\Delta t}')^2\sigma_{ax}^2\Delta t^2)}{\lambda^i}) \tag{14}$$

$$Var\left(Y_n'\right) = \sigma_p^2 + \sum_{i=1}^{n}(\frac{(Vx_{i\Delta t}'\Delta t)^2}{\lambda^i}(\sigma_\theta^2 + \sum_{j=1}^{i}\left(Yawrate_t'\right)^2\sigma_{yawrate}^2\Delta t^2)) \tag{15}$$

where λ is a time attenuation coefficient, which is used to gradually attenuate the influence of velocity fluctuation after a period of time. For a single obstacle, it can be expressed as a two-dimensional Gaussian distribution with covariance $\begin{pmatrix} Var\left(X_n'\right) & 0 \\ 0 & Var\left(Y_n'\right) \end{pmatrix}$ in the Frenet coordinate system whose reference path is predicted trajectory. However, when there are too many obstacles in the environment, projecting the location points of the ego vehicle into multiple prediction trajectory coordinate systems will greatly increase the computational requirements. Therefore, the above distribution should be approximated to the road coordinate system. Here, Gaussian distribution above is approximated to a two-dimensional Gaussian distribution considering the heading angle in the road coordinate system. Consequently, the location distribution under the road coordinate system can be expressed as:

$$\begin{pmatrix} \mathbf{X}_n \\ \mathbf{Y}_n \end{pmatrix} \sim \mathcal{N}(\mu_n, D_n) \tag{16}$$

$$\mu_n = \begin{pmatrix} X_{n\Delta t} \\ Y_{n\Delta t} \end{pmatrix} \tag{17}$$

$$D_n = R_n^{\mathrm{T}} \begin{pmatrix} Var\left(X_n'\right) & 0 \\ 0 & Var\left(Y_n'\right) \end{pmatrix} R_n \tag{18}$$

$$R_n = \begin{pmatrix} \cos(\theta_{n\Delta t}) & -\sin(\theta_{n\Delta t}) \\ \sin(\theta_{n\Delta t}) & \cos(\theta_{n\Delta t}) \end{pmatrix} \tag{19}$$

Once the track distribution of each obstacle is obtained in turn, the risk level of the vehicle in the future can be evaluated.

4.3. Collision probability assessment

The collision risk of the ego vehicle is computed based on the collision probability between the main vehicle and each obstacle and assume that there are several obstacles in the environment. Firstly, we compute the random position distribution of all obstacles according to the method presented in Section 4.2. Then, we evaluate the probability of collision with each obstacle respectively. Taking the nth sampling point (I, J) as an example, the rigorous computation of collision probability risk is represented by:

$$P_{nm} = \iint_D f(x,y)dxdy \tag{20}$$

where D is the rectangular area of the vehicle outline with discrete sampling points serving as the centroid, and $f(x,y)$ is the probability density function of Gaussian distribution. However, the complexity of this integral computation is relatively high, since a large number of double integral computations will be carried out when there are many obstacles in the environment. Because the change of probability density in the rectangular area is usually small, the collision probability can be evaluated by:

$$P_{nm} = S_D \max(f(M)) \tag{21}$$

$$M = \begin{cases} (i,j) \\ (i - bsin\theta, j + bcos\theta) \\ (i + acos\theta, j + asin\theta) \\ (i - acos\theta, j - asin\theta) \\ (i + bsin\theta, j - bcos\theta) \\ (i + \begin{bmatrix} sin\theta & cos\theta \end{bmatrix} \begin{bmatrix} -b \\ a \end{bmatrix}, j + \begin{bmatrix} cos\theta & sin\theta \end{bmatrix} \begin{bmatrix} b \\ a \end{bmatrix}) \\ (i + \begin{bmatrix} sin\theta & cos\theta \end{bmatrix} \begin{bmatrix} b \\ a \end{bmatrix}, j + \begin{bmatrix} cos\theta & sin\theta \end{bmatrix} \begin{bmatrix} -b \\ a \end{bmatrix}) \\ (i + \begin{bmatrix} sin\theta & cos\theta \end{bmatrix} \begin{bmatrix} -b \\ -a \end{bmatrix}, j + \begin{bmatrix} cos\theta & sin\theta \end{bmatrix} \begin{bmatrix} b \\ -a \end{bmatrix}) \\ (i + \begin{bmatrix} sin\theta & cos\theta \end{bmatrix} \begin{bmatrix} b \\ -a \end{bmatrix}, j + \begin{bmatrix} cos\theta & sin\theta \end{bmatrix} \begin{bmatrix} -b \\ -a \end{bmatrix}) \end{cases} \tag{22}$$

where S_D is the rectangular area of the vehicle outline, θ is the heading angle of the ego vehicle, a and b are the half length and half width of the vehicle. It can be seen that the product of the maximum probability density of nine sampling points of the vehicle and the rectangular area of the vehicle outline is taken as the approximate value of collision probability. Finally, after calculating the collision probability with all obstacles according to the above method, the total collision probability is computed through Bayesian equation as:

$$\mathrm{R\_sum}_{ij} = 1 - \prod_{\mathrm{m}=1}^{\mathrm{ob\_n}}(1 - \mathrm{p}_{nm}) \tag{23}$$

where ob_n is the number of obstacles in the environment. The collision with each obstacle can be regarded as independent, so the total collision probability of the nth sampling point is only related to each independent collision probability. In the following, the total collision probability will be used to describe the risk level of the sampling point.

5. Spatio-temporal planning

In this section, we propose a layered trajectory planning method which combines probabilistic risk assessment and reachable sets. In rough planning, a risk-sensitive set partition method is proposed to handle the difficulty of setting risk classification by now. Then the subset multi-tree structure is constructed according to the kinematics model, and the optimal subset chain is solved by dynamic programming (DP). At the fine planning level, a QP method is used to optimize the solution of the trajectory in the driving corridor.

5.1. Reachable sets considering risk level

In vehicle dynamic modeling, the longitudinal and lateral motions of the vehicle can be treated to be independent of each other. Therefore, the two-dimensional reachable set is first divided into two one-dimensional reachable sets, $Sx(s,vx,t) \in \boldsymbol{Reachx}\left(T;S_0\right)$ and $Sy(p,vy,t) \in \boldsymbol{Reachy}\left(T;S_0\right)$, which can be separately computed. Ultimately, these two sets are integrated to construct the final reachable set.

Take evaluating the longitudinal one-dimensional reachable set as an example: at the initial time, the vehicle state is represented to be a point in the $s-vx$ plane. Starting from this point, the position that the vehicle can reach after a time interval Δt is a convex set due to the continuity of motion. According to Sontges' work, the upper and lower boundaries of each convex set can be obtained by:

$$s = s_0 + vx_0\Delta t + ax_{min}\Delta t^2\left(r - \frac{1}{2}r^2\right) + ax_{max}\Delta t^2\left(\frac{1}{2} - r + \frac{1}{2}r^2\right)$$

$$vx = vx_0 + ax_{min}\Delta t + ax_{max}(1-r)\Delta t \tag{24}$$

where r denotes the range of a convex set which is set between [0:1], ax_{max} and ax_{min} are respectively the maximum and minimum longitudinal accelerations of the vehicle, $r\Delta t$ represents the deceleration period, and $(1-r)\Delta t$ denotes the acceleration period. By discretizing r, we can adopt a series of (s,vx) to be the upper boundary points of convex sets. Similarly, the lower boundary can be obtained by:

$$s = s_0 + vx_0\Delta t + ax_{max}\Delta t^2\left(r - \frac{1}{2}r^2\right) + ax_{min}\Delta t^2\left(\frac{1}{2} - r + \frac{1}{2}r^2\right)$$
$$vx = vx_0 + ax_{max}\Delta t + ax_{min}(1-r)\Delta t \tag{25}$$

And then, the set of motion states that the vehicle can reach at t_1 are obtained. In order to reduce the computation cost, we exploit the convex polygon based on several state points on the boundary as the approximation of the original convex set. In the light of previously discussed method, the convex sets based on all boundary points at time step t_n are evaluated and the Minkowski sum of these convex sets is computed to acquire the reachable set $\boldsymbol{Reachx}\left(T;S_0;t_{n+1}\right)$ at time t_{n+1}.

Similar to the evaluation in longitudinal dimension, the reachable set in lateral dimension can also be evaluated. After the ideal reachable sets in two directions are determined, they are required to be tailored under the constraints of vehicles and roads. We assume the longitudinal speed to be nonnegative and not greater than the speed limits of the road vx_{max}. Meanwhile, the lateral position needs to be restricted within the road boundary.

To sum up, for a two-dimensional ideal reachable set $\boldsymbol{Reach'}(T;S_0)$ without considering obstacles, it can be evaluated by the Cartesian product of the longitudinal and lateral reachable sets:

$$\begin{aligned}\boldsymbol{Reach'}\left(T;S_0\right) = \{S\left(s_n,p_n,vx_n,vy_n,t_n\right)t_n \in [0,T]\,|s_n,\\ vx_n \in \boldsymbol{Reachx}\left(T;S_0;t_n\right);p_n,vy_n \in \boldsymbol{Reachy}\left(T;S_0;t_n\right)\}\end{aligned} \tag{26}$$

After obtaining the ideal two-dimensional reachable set, the unreachable set is removed based on the risk and the projection of reachable set in the $s-p$ plane. The traditional reachable set algorithm in literature (Manzinger et al., 2021) divides the reachable set into multiple convex sets after removing the occupied area by obstacles, which will cause each convex set to have different sizes after segmentation. At the same time, the convex set with too large area will lead to excessive differences in internal risk levels, which makes it difficult to evaluate the safety and driving efficiency of each set. Therefore, we propose a method to segment subsets according to the risk level, which can produce more accurate representation of the risk value and connection relationship of the set. The steps to segment the reachable subsets are shown in Fig. 6.

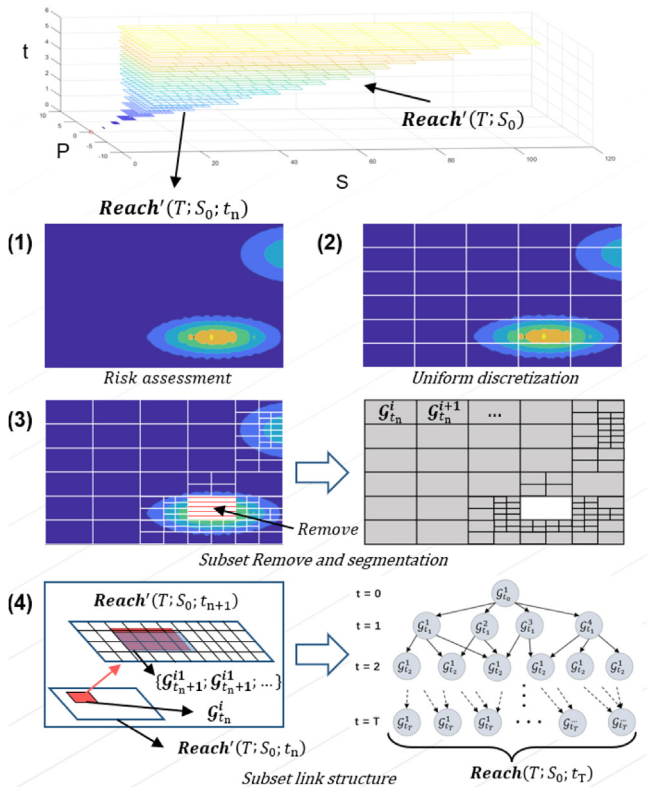


Fig. 6. Segmentation steps of reachable subsets based on risk level: In the s-p-t coordinate system at the top of the figure is the forward reachable set $\boldsymbol{Reach'}\left(T;S_0\right)$ derived earlier. The following (1), (2), (3), (4) represent the four steps of Segmentation, which are explained in more detail below.

The ideal set by the risk distribution of set projection is segmented in the condition that the resulting subsets are convex sets. We denote the ith subset of t_n as $\mathcal{G}^i_{t_n}$ and establish a one-way linked list to represent the link relationship of subsets between adjacent times:

$$\left\{\mathcal{G}^i_{t_n} \rightarrow \left(\mathcal{G}^j_{t_{n+1}},\mathcal{G}^k_{t_{n+1}}\ldots\mathcal{G}^l_{t_{n+1}}\right) \in \boldsymbol{Reach'}\left(T;S_0;t_{n+1}\right)\right\} \tag{27}$$

That is to say, when the vehicle state $S(s,p,vx,vy,t_n) \in \mathcal{G}^i_{t_n}$, for any $son \in \{j,k\ldots l\}$ there is at least one set of inputs (ax,ay) such that at t_{n+1} vehicle state $S'(s,p,vx,vy,t_{n+1}) \in \mathcal{G}^{son}_{t_{n+1}}$. The computation steps of the link relationship between subsets are listed as follows:

① ***Risk assessment:*** The ideal reachable set at each time is uniformly sampled in the projection area of the $s-p$ plane to obtain dense sampling points. Combined with the obstacle prediction trajectory, the risk level of each sampling point is evaluated (see in Section 2). Then the total risk level situation in the projection area can be obtained.

② ***Uniform discretization:*** As the preprocess of subset segmentation, the projection area is divided into uniform grids. All state points of projection points in the same uniform grid are based on the same subset, which is denoted as $\mathcal{G}b^i_{t_n}$. For the position coordinates of the grid center point, in order to represent the position characteristics of the basic subset, it is necessary to limit the size of the grid. We set the grid length as $\min((0.2+0.1n), L_{t_n}/5)$, and width as $\min(0.5, D_{t_n}/5)$,

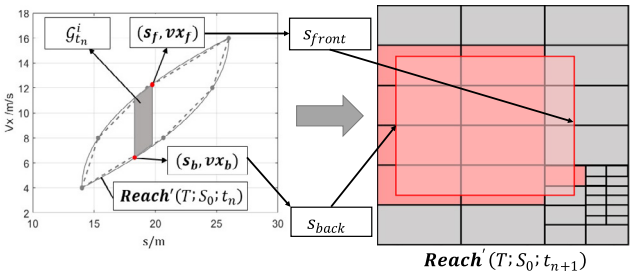


Fig. 7. Subset linking process: The pink rectangle denotes the reachable area of subset $\mathcal{G}^i_{t_n}$ at the next time step, and the red rectangles are the children subset of subset $\mathcal{G}^i_{t_n}$.

where n denotes the nth time step, L_{t_n} and D_{t_n} are the length and width of the projection area of $\boldsymbol{Reach'}(T; S_0; t_n)$ respectively.

③ ***Subset Remove and segmentation:*** According to the risk distribution, the grid is further divided. If the maximum difference of risk values of sampling points in the grid is less than the threshold value P_{av}, the average risk level in the grid can be adopted to represent most of state points in the grid and no further segmentation is required. The grids that do not meet the judgment are quartered until the maximum difference of risk values is lower than P_{av}. After the division, if the average risk level $P^m_{(i,t_n)}$ becomes less than the safety risk threshold P_{th}, the grid will be marked as low-risk subset $\mathcal{G}^i_{t_n}$. If it is higher than P_{th}, it will be marked as $\mathcal{G}hr^i_{t_n}$.

④ ***Subset link structure:*** After obtaining the low-risk subset set at each time, the children subset of each subset $\mathcal{G}^i_{t_n}$ is evaluated. First, we compute the reachable area of subset $\mathcal{G}^i_{t_n}$ at the next time step. The boundary of the reachable area is calculated separately in longitudinal and lateral dimension. Take the longitudinal as an example: the longitudinal front and rear boundaries of the reachable area are computed according to the front and rear endpoints of the subset, as shown in Fig. 7. The rear boundary of the reachable area s_{back} is the longitudinal position after state $(\boldsymbol{s_b}, \boldsymbol{vx_b})$ decelerates for a time step Δt with deceleration ax_{min}:

$$s_{back} = s_b + vx_b \Delta t + ax_{min} \Delta t^2 \tag{28}$$

And the front boundary of the reachable area s_{front} is the longitudinal position $(\boldsymbol{s_f}, \boldsymbol{vx_f})$ after accelerating for a time step Δt with acceleration ax_{min}:

$$s_{front} = s_f + vx_f \Delta t + ax_{max} \Delta t^2 \tag{29}$$

The lateral computation method is the same as the above method. After obtaining the reachable area. for each grid at time t_{n+1} we compute the ratio of the area intersected with the reachable area to the total area. We mark the corresponding subset of grids whose area ratio is more than 80% as $\mathcal{G}^i_{t_n}$'s linked subset.

It should be noted that the subsets and link relationship evaluated in this way are not completely accurate, for the state deviation on the velocity component is ignored in the process. However, these deviations can be avoided in the subsequent detailed planning. So far, we can adopt a multi-tree structure with subsets as nodes from t_0 to t_N. The set of subsets in this multi-tree structure is denoted as $\boldsymbol{Reach}(T; S_0)$, and the subsets with the same element t_n are marked as $\boldsymbol{Reach}(T; S_0; t_n)$.

5.2. Rough search based on dynamic programming

After the reachable sets and the multi-tree structure are evaluated, we need to search a subset chain from t_0 to t_N. Different from heuristic search algorithms requiring a search target point in advance (Petereit et al., 2012), DP, a widely used search algorithm, can be exploited to realize the rough search (Jia et al., 2020; Wei et al., 2017). Moreover, since the multi-tree structure we used takes the state set as the node, its number of nodes is far less than that of the traditional spatio-temporal map, thus effectively reducing the search time of DP algorithm.

Firstly, we compute the cost of each node considering the efficiency, safety and road centrality of the node subset. The cost function $G\left(\mathcal{G}^i_{t_n}\right)$ is represented by:

$$G\left(\mathcal{G}^i_{t_n}\right) = C_e f_e\left(\mathcal{G}^i_{t_n}\right) + C_r f_r\left(\mathcal{G}^i_{t_n}\right) + C_c f_c(\mathcal{G}^i_{t_n}) \tag{30}$$

where C_e, C_r and C_c are the weight coefficients, respectively. $f_e\left(\mathcal{G}^i_{t_n}\right)$ is the driving efficiency cost obtained by:

$$f_e\left(\mathcal{G}^i_{t_n}\right) = \sqrt{n^2+1}(X^{front}_{t_n} - x_{(i,t_n)})/L_{t_n} \tag{31}$$

where $X^{front}_{t_n}$ denotes front boundary of the projection area's in $\boldsymbol{Reach'}(T; S_0; t_n)$; L_{t_n} is the length of $\boldsymbol{Reach'}(T; S_0; t_n)$; $x_{(i,t_n)}$ is the ordinate of the central point of subset $\mathcal{G}^i_{t_n}$, $\sqrt{n^2+1}$ is the time weighting coefficient, which is adopted to improve the proportion of long-term time efficiency and to avoid the pursuit of short-term efficiency.

The safety cost $f_r\left(\mathcal{G}^i_{t_n}\right)$ can be calculated as follows:

$$f_r\left(\mathcal{G}^i_{t_n}\right) = R\_sum^m_{(i,t_n)}/S_{(i,t_n)} \tag{32}$$

where $R\_sum^m_{(i,t_n)}$ is the average risk level in the grid $\mathcal{G}^i_{t_n}$ and $S_{(i,t_n)}$ is the grid area.

The cost of the road center $f_c\left(\mathcal{G}^i_{t_n}\right)$ which makes the vehicle to keep driving on the center line of the road, is represented by:

$$f_c\left(\mathcal{G}^i_{t_n}\right) = -\cos\left(\frac{2\pi}{L_{road}\, y_{(i,t_n)}}\right) + 1 \tag{33}$$

where L_{road} is the road width, $y_{(i,t_n)}$ is the abscissa value of the central projection point of subset $\mathcal{G}^i_{t_n}$, and the cos function is used for fitting.

After attaining the cost of each node, DP is utilized to solve the best subsets chain based on multi-tree structure. In dynamic programming, we establish a storage structure $\boldsymbol{DP}$ with the same structure as the subset multi-tree to store the optimal solution of the branch subproblem. By solving the minimum subproblem from bottom to top, we can obtain the subset chain that minimizes the total cost, which is expressed as $\{\mathcal{G}^{i0}_{t_0} \to \mathcal{G}^{i1}_{t_1} \to \cdots \to \mathcal{G}^{iT}_{T}\}$. The process of DP is described in Algorithm 2. By Algorithm 1, an optimal solution of subset chain $\{\mathcal{G}^{i0}_{t_0} \to \mathcal{G}^{i1}_{t_1} \to \cdots \to \mathcal{G}^{iT}_{T}\}$ from t_0 to t_N can be found to minimize the total cost from t_0 to t_N. Finally, the longitudinal and lateral coordinates of the projection center of each optimal subset are marked as the optimal trajectory points (X_x, Y_r).

5.3. Node expansion

After obtaining the optimal subset chain, as the projection area of the subset is small, it is not suitable to be used as the boundary of subsequent trajectory optimization. Based on the optimal subset, the collision-free area needs to be expanded to become larger. At the same time, the area should be convex to ensure an optimal solution.

Here, the solution of convex polygon is simplified to search the maximum low-risk rectangular at each time step. In order to prevent the expansion area from being narrow, the expansion is executed in three steps. Firstly, to ensure the minimum size of the longitudinal and lateral boundary, the subset is expanded to the largest square centered on the optimal point. Then, the square is expanded horizontally and vertically. The expansion steps are shown in Fig. 8. After obtaining the extended area at each time step, a driving corridor from t_0 to t_N is constructed for subsequent trajectory optimization.

5.4. Trajectory optimization

After obtaining the driving corridor, the sequential quadratic programming method is implemented to solve the optimal trajectory. In trajectory optimization, longitudinal optimization is first computed and

Algorithm 2 : dynamic programming for subset Link structure search

Input: A subset Link structure S ; The subset cost $G(\mathcal{G}_{t_n}^i)$;

Output : A $Chain$ set of subset from t_0 to T;

1. **Initialize** the $Chain$ set with $\{\mathcal{G}_{t_0}^0 \rightarrow\}$、the DP-table structure $\boldsymbol{DP}$ with subset Link structure ;
2. **Initialize** $\boldsymbol{DP}$ nodes in time step T $\{dpnode_T^0; \ldots; dpnode_T^i\}$ with $\{G(\mathcal{G}_T^0); \ldots; G(\mathcal{G}_T^i)\}$
3. **For** i = $T-1$: t_0
4. **For** each subset $\mathcal{G}_{t_i}^n$ in time step t_i **do**
5. Find Children subset $\{\mathcal{G}_{t_{i+1}}^j \ldots \mathcal{G}_{t_{i+1}}^k \ldots \mathcal{G}_{t_{i+1}}^l\}$ of $\mathcal{G}_{t_i}^n$;
6. Find minimum cost node $dpnode_{i+1}^{nmin}$ in $\{dpnode_{i+1}^j \ldots dpnode_{i+1}^l\}$;
7. $dpnode_{t_i}^n = dpnode_{i+1}^{nmin} + G(\mathcal{G}_{t_i}^n)$;
8. Mark $\mathcal{G}_{t_{i+1}}^{nmin}$ is the minimum Children set of $\mathcal{G}_{t_i}^n$
9. **End For**
10. **End For**
11. **While** $Chain$ set's last subset is not in time step T **do**
12. Find last subset of $Chain$;
13. Find last subset's minimum Children set;
14. Add Children set to $Chain$ set
15. **End While**
16. **Return** $Chain$ $\{\mathcal{G}_{t_0}^{i0} \rightarrow \mathcal{G}_{t_1}^{i1} \rightarrow \cdots \rightarrow \mathcal{G}_T^{iT}\}$

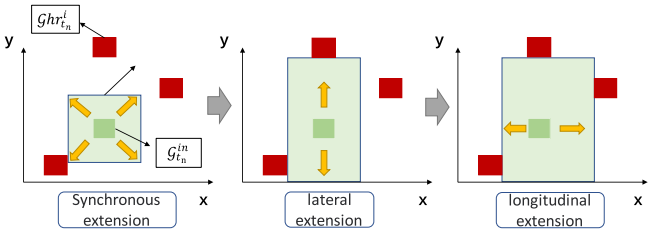


Fig. 8. Steps of node expansion.

the lateral optimization will be the next. The optimization indexes include longitudinal acceleration, longitudinal jerk and longitudinal distance relative to the optimal node. The cost function $f(x)$ can be expressed as:

$$f(x) = C_1 {A_x}^2 + C_2 {J_x}^2 + C_3 \left(X_x - x\right)^2 \tag{34}$$

$$\text{s.t. } x \geq x_{min}, x \leq x_{max} \tag{35}$$

$$A_k = \frac{x_k - 2x_{k-1} + x_{k-2}}{\Delta t^2} \tag{36}$$

$$J_k = \frac{x_k - 3x_{k-1} + 3x_{k-2} - x_{k-3}}{\Delta t^3} \tag{37}$$

where C_1, C_2, C_3 are the weight coefficients, x_{min}, x_{max} are the longitudinal boundary of the driving corridor, A_k is the cost of longitudinal acceleration at time step t_k, A_x is the total acceleration cost accumulated from t_0 to t_N, J_k is the cost of longitudinal jerk at time step t_k, J_x is the overall jerk cost accumulated from t_0 to t_N and x_k is the longitudinal optimal point at time t_k;

In lateral optimization, the optimization indexes include vehicle lateral speed, path curvature and lateral distance from the optimal node. The cost function $f(y)$ is expressed as:

$$f(y) = C_4 {A_y}^2 + C_5 {C_{ky}}^2 + C_6 \left(y_y - y_r\right)^2 \tag{38}$$

$$\text{s.t. } y \geq y_{min}, y \leq y_{max} \tag{39}$$

(39)

$$A_{yk} = \frac{y_k - 2y_{k-1} + y_{k-2}}{\Delta t^2} \tag{40}$$

$$C_k = \frac{2\left(\frac{y_{k+1}-y_k}{x_{k+1}-x_k} - \frac{y_k-y_{k-1}}{x_k-x_{k-1}}\right)}{x_{k+1} - x_{k-1}} \tag{41}$$

where C_4, C_5, C_6 are the weight coefficients, y_{min}, y_{max} are the lateral boundary of the driving corridor, A_{yk} is the cost of lateral acceleration at time step t_k, A_x is the total acceleration cost accumulated from t_0 to t_N, C_k is the cost of path curvature at time step t_k, J_x is the overall curvature cost accumulated from t_0 to t_N and y_k is the lateral optimal point at time t_k.

6. Experiments and results

The real-time simulation platform is a driving simulator based on PXI real-time system. As shown in Fig. 9, the hardware component includes a chassis system of a car, a lower computer PXI, the upper PC-I with the communication framework and planning algorithm, and the upper PC-II for test scene simulation. In the upper PC-I, the proposed programming algorithm is compiled in LabVIEW-RT. In addition, based on the NI/PXI real-time platform, the real-time vehicle dynamics model is built by CarsimRT. In the upper PC-II, the traffic scene and the sensor model are established through Prescan, and the perceptual information is transmitted to the obstacle prediction model running in Python. UDP is used for the communication between two upper PC, and the upper PC-I can communicate with PXI through TCP/IP.

The ego vehicle in Prescan deploys typical sensors to simulate the perception blind zone and perception error in the real traffic. The sensors shown in Fig. 11 including one lidar with 360° FOV, one front camera with 120° FOV, and one front radar with 90° FOV are equipped

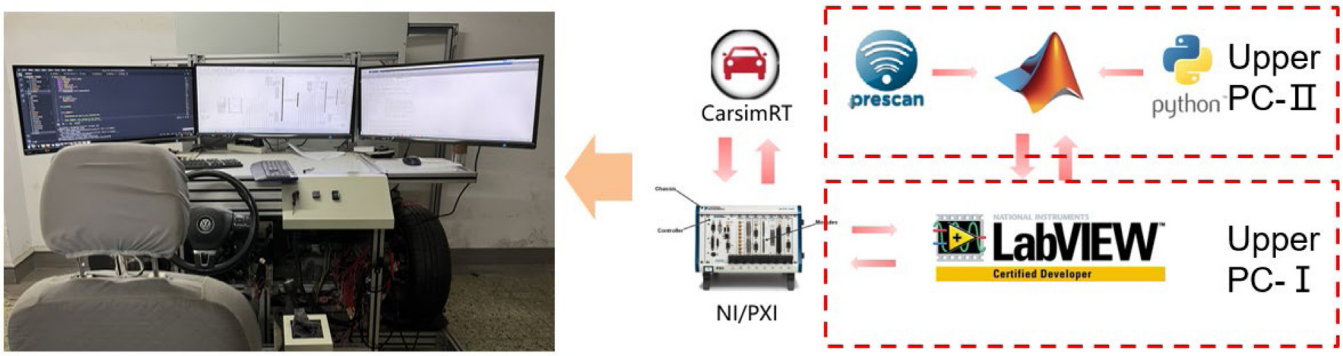


Fig. 9. HIL Platform.

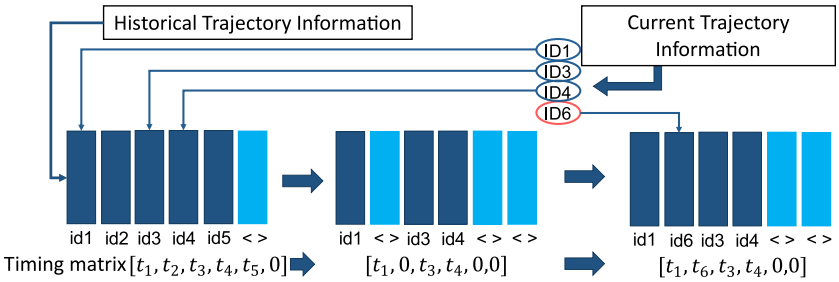


Fig. 10. Obstacle screening before prediction.

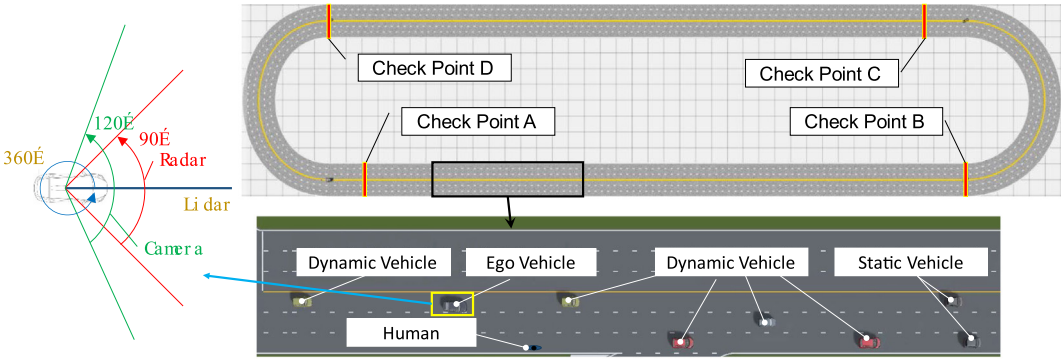


Fig. 11. Circular test scenario and ego vehicle perception system.

to perceive the surrounding vehicles and pedestrians. At the same time, the historical information of perceptible surrounding obstacles is recorded, which is input to the GNN-LSTM prediction network for trajectory prediction as described in Section 4. The detailed process is shown in Fig. 10.

Since the LSTM network requires a continuous period of historical tracks, we need to judge and store the perceptual data which have enough historical time. It can be divided into three steps: First, a trajectory matrix and a historical time vector are maintained. In this paper, their dimension are both six. When the current perceptual data is obtained, the vehicle information that exists in both the matrix and the perceptual data is updated in trajectory matrix. Secondly, the trajectory data in matrix which do not exist in the perceptual data are deleted and the historical time vectors are updated. Finally, data that exists in the perceptual data but is not in the matrix is added to the matrix. This allows real-time filtering and extraction of historic tracks that meet the requirements for prediction.

6.1. Random scenario test

To verify the generalization ability and robustness of this algorithm, we build a circular test scenario in Prescan, which can randomly generate surrounding vehicles and pedestrians during the test. In this scenario shown in Fig. 11, the test vehicle will encounter a large number of random traffic scenarios with randomly generated vehicles and pedestrians. The long-time automatic testing can help us to evaluate the algorithm performance in different traffic scenes. The test scenario is a circular road with a length of 1025 m, in which the straight road is 750 m with the speed limit 15 m/s, and the length of the curve is 275 m with the speed limit 8 m/s. Four checkpoints are set out on the road. When the ego vehicle reaches checkpoints A and C, nine dynamic vehicles, two static vehicles and one pedestrian are generated on the current road segment. Static vehicle position coordinates are randomly distributed variables. And there is a 25% probability that static vehicles ignore the lane line to park on it. Pedestrians are created in a fixed position and will cross the road at a speed of 1 m/s. Generation location, initial speed, and expected speed of dynamic vehicles are all randomly distributed.

Unlike the fixed trajectory of surrounding vehicles in the released dataset tests (Mu et al., 2021), the dynamic vehicles in this test are interactive to use the network to perceive information about surrounding vehicles and pedestrians. The lane-changing decision and speed planning are determined by MOBIL and IDM models (Kesting et al.,

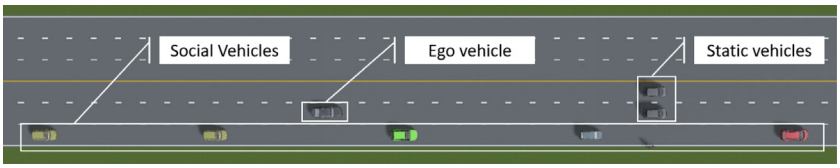


Fig. 12. Corner case in random test.

Table 4
Random variable distribution of test scenario.

Parameter	Random distribution
Vehicle initial state	$\mathrm{U}(60,400)\,(check\ point\ A)/\mathrm{U}(550,890)\,(check\ point\ C)$
Vehicle initial velocity	$\mathrm{U}(6,15)$
Vehicle desired velocity	$\mathrm{U}(10,15)$
Politeness factor p	$\mathcal{N}(0.5,0.5^2)$
Safety time gap T_{dm}	$\mathcal{N}(1.2,0.8^2)$
Threshold value Δa_{th}	$\mathcal{N}(2,0.5^2)$

Table 5
Main parameter setting.

Parameter	Value	Parameter	Value
Δt	0.3 (s)	C_c	0.4
T	6 (s)	C_1	5.8
ax_{max}/ax_{min}	+5/−5 (m/s^2)	C_2	1.2
ay_{max}/ay_{min}	+5/−5 (m/s^2)	C_3	7.8
vx_{max}	15 (m/s)	C_4	4.2
C_e	1.8	C_5	1
C_r	650	C_6	2.2

2007; Treiber et al., 2000), and the lane-changing trajectory is planned by quintic polynomial curve. The longitudinal and lateral motions are controlled by sliding mode algorithm. The MOBIL and IDM models can be developed by:

$$IDM\ model: \quad \begin{aligned} \dot{v_\alpha} &= a_{dm}\left[1-\left(\frac{v_\alpha}{v_{road}}\right)^4-\left(\frac{s_*(v_\alpha,\Delta v_\alpha)}{s_\alpha}\right)^2\right] \\ s_*\left(v_\alpha,\Delta v_\alpha\right) &= s_0+v_\alpha T_{dm}+\frac{\Delta v_\alpha}{2\sqrt{a_{dm}b_{dm}}} \end{aligned} \tag{42}$$

$$MOBIL\ model: \quad \widetilde{a_c}-a_c+p\left(\widetilde{a_n}-a_n+\widetilde{a_o}-a_o\right)>\Delta a_{th} \tag{43}$$

In IDM model, $\dot{v_\alpha}$ is the expected acceleration; v_α is the current speed; v_{road} is the road speed limit; s_α is the distance from the front car; T_{dm} is the time headway (TH) of vehicle; a_{dm} and b_{dm} are the maximum lateral and longitudinal acceleration of the vehicle. In MOBIL model, p is a parameter to characterize the degree of selflessness of drivers; a_c, a_n and a_o are the expected accelerations of the ego vehicle, rear vehicle in the target lane and in the current lane, after lane change; $\widetilde{a_c}$, $\widetilde{a_n}$ and $\widetilde{a_o}$ are the expected accelerations of the ego vehicle, rear vehicle in the target lane and the current lane, before lane change. Δa_{th} is the switch threshold. All the above expected accelerations are computed based on IDM model. To enhance the randomness of the test scenario, the parameters p, Δa_{th}, and T_{dm} in the two models, which characterize driving style, are set as random variables subject to different Gauss distributions. Also, the driving styles of the generated surrounding vehicles are randomly constructed. The distributions of random parameters for this test scenario are listed in Table 4. And Table 5 shows the key parameter settings in the algorithm during the test

We compare our method with the reachable set planning algorithm in Sontges and Althoff (2018) (denoted as RS+QP) and the decision and planning method based on MOBIL and IDM in Kesting et al. (2007). In addition, to verify the accuracy of the prediction algorithm proposed in Section 4, the CTRA prediction method are adopted for comparison. All four algorithms are run for 5000 s in a test scenario. To evaluate the efficiency, safety and comfort of the algorithm, the average test results per 1000 s are recorded to include average speed, the number of dangerous behavior and longitudinal acceleration RMS. The driving behaviors that result in emergency braking of ego vehicle or surrounding vehicles are treated as dangerous behaviors of corner case.

Table 6
The statistical results in random test scenario.

Method	Average speed (m/s)	Dangerous behavior (times)	ax RMS (m/s^2)
MOBIL+IDM	8.823	3.6	2.398
RS+QP	9.281	1.5	2.8405
Our method with CTRA	10.804	1.8	2.0561
Our method	10.218	0.8	1.6986

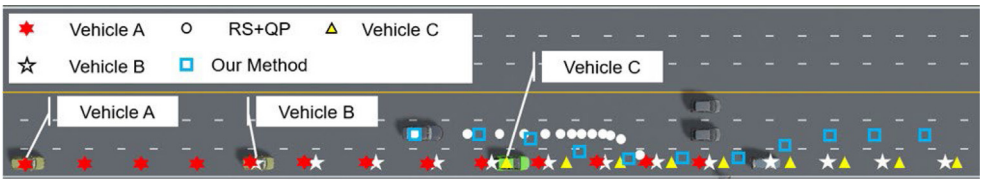
From Table 6, the statistical results show that the driving efficiency of the proposed algorithm is 10.9% higher than that of RS+QP method in 1000 s test time. Compared with RS+QP method, the number of dangerous behaviors is decreased by 46%; The longitudinal acceleration RMS is also reduced by 29.16% compared with MOBIL+IDM, which performs better than other control method. When compared with CTRA control algorithm, it can be found that due to the same planning algorithm, the difference between driving efficiency and drive comfort is not obvious. However, due to the more accurate prediction of GNN-LSTM, the number of dangerous behaviors is reduced one time per 1000 s on average, which proves the accuracy of the prediction algorithm. The above statistical data also testify the reliability and robustness of this algorithm in high traffic density scenarios.

6.2. Corner case

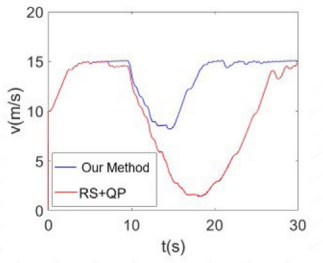
After the random test, a merge scenario is constructed for corner case testing since it is easy to bring about planning failures during testing. As shown in Fig. 12, the scene is a three-lane road with a width of 3.5 m. The ego vehicle is controlled by the trajectory planning algorithm and its expected speed is 15 m/s. The static vehicles are parked side by side to only allow the traffic to go through on the third lane, where there are five dynamic vehicles with an initial 20 m inter-vehicle distance to travel at a desired speed of 8 m/s. Dynamic vehicles based on IDM model do not change lanes at all times, but can autonomously control vehicle speed according to the state of the front vehicle. In this scenario test, our proposal is compared with RS+QP (Sontges and Althoff, 2018).

The simulation results are shown in Fig. 13. It can be seen that the two methods choose different driving behaviors during the test in Fig. 13(a). By the competitor RS+QP, the ego vehicle does not immediately choose to merge into the traffic flow on the third lane, but parks at the current lane to wait for the merging. Instead, by our method the ego vehicle first decelerates to merge into the third lane, and then accelerates to drive out of the slow-speed traffic flow after it crosses over the static obstacles. And Fig. 13(b), (c), (d) also shows the results of the trajectory points by the two methods, and illustrates a comparable result of the vehicle dynamic driving data. Compared with the control method RS+QP, the improvements by our proposal can be found: the speed loss of the whole process is reduced by 47%, the lateral and longitudinal acceleration is smoother, and the passing time is reduced by 8s.

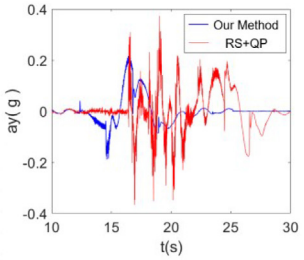
For RS+QP, the ego vehicle decelerates when static obstacles are detected. However, instead of slowing down to merge into traffic flow,



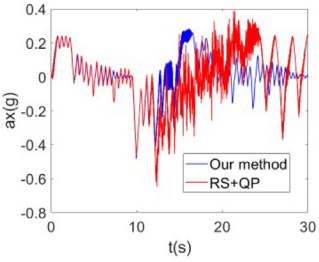
(a) Vehicle trajectory result of corner case



(b) Comparison results of longitudinal speed

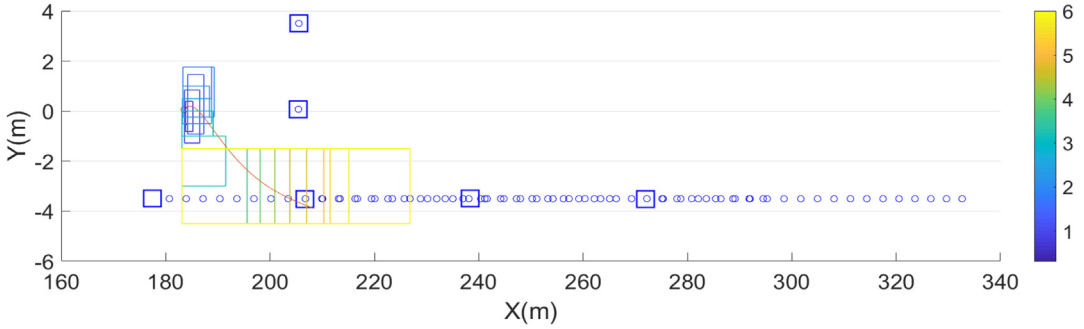


(c) Comparison results of lateral acceleration

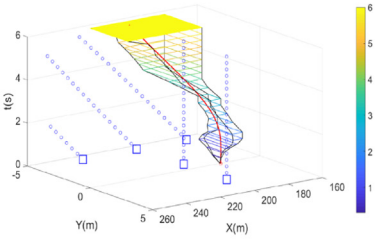


(d) Comparison results of longitudinal acceleration

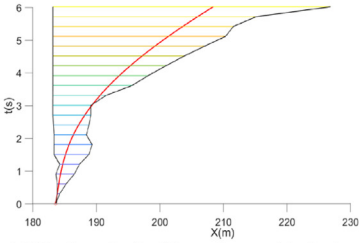
Fig. 13. Test results of corner case.



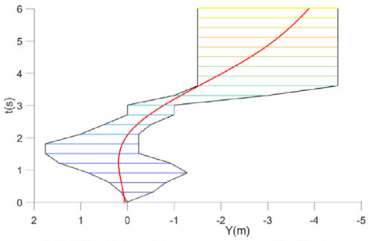
(a) RS+QP planning trajectory and drivable areas at time step 18s



(b) RS+QP driving corridor result



(c) The longitudinal boundary and trajectory



(d) The lateral boundary and trajectory

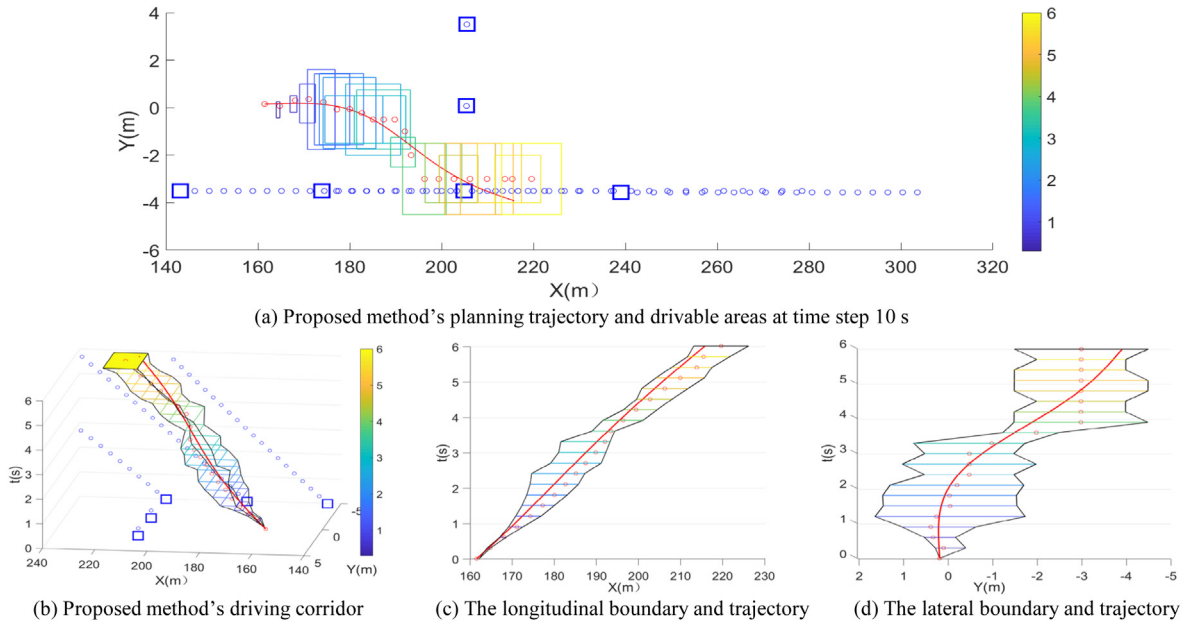
Fig. 14. RS+QP planning results and drivable areas at time step 18 s, In (a), the rectangle represents the planned driving area at each time, and the color band on the right represents the border color corresponding to the driving area at each preview time, In (b), (c), (d), the red curve represents the last optimized smooth track, The black polyline indicates the boundary of the driving corridor.

it chooses a corridor with a larger drivable area for lane keeping. When its speed is reduced to less than 5 m/s, the ego vehicle loses the speed condition to cut into the traffic flow. Therefore, the ego vehicle can only stop and wait for other vehicles to change the lane after passing. Fig. 14(a–d) show the driving corridors generated by the RS+QP method in time step 18 s. Although the driving corridor boundaries of this method are wider, unreasonable risk assessment will lead to conservative driving behavior.

For the proposed method, the ego vehicle first slows down after it senses static obstacles. When its speed drops to be close to that of the target traffic flow, the algorithm evaluates a safe driving corridor that allows the ego vehicle to merge safely. Also, the ego vehicle drives out of the low-speed traffic flow to change lane once crossing over the static obstacles.

The results of the corridor and trajectory at time step 10 s are shown in Fig. 15. It can be seen that the corridor controls the speed before the merging, so that the vehicle will not accelerate or decelerate significantly during the lane-changing process. Fig. 15(a) illustrates the top view of the drivable area and the optimized trajectory at time step 10 s. It also shows the lateral and longitudinal boundaries of the driving corridor, as well as the final trajectory. Fig. 16 shows the detailed results at time step 12 s. In that step, after crossing the static obstacles, the ego vehicle drives out of low-speed traffic flow without long following. Therefore, the trajectory in the third lane is not entirely extended to the center of the road, but is closer to the middle lane to change lanes faster after crossing the static obstacles, which is more similar to the driving habits of human drivers. Therefore, our proposed method can better assess the risk level of each state, and also balance driving efficiency and driving risk complex traffic environment.

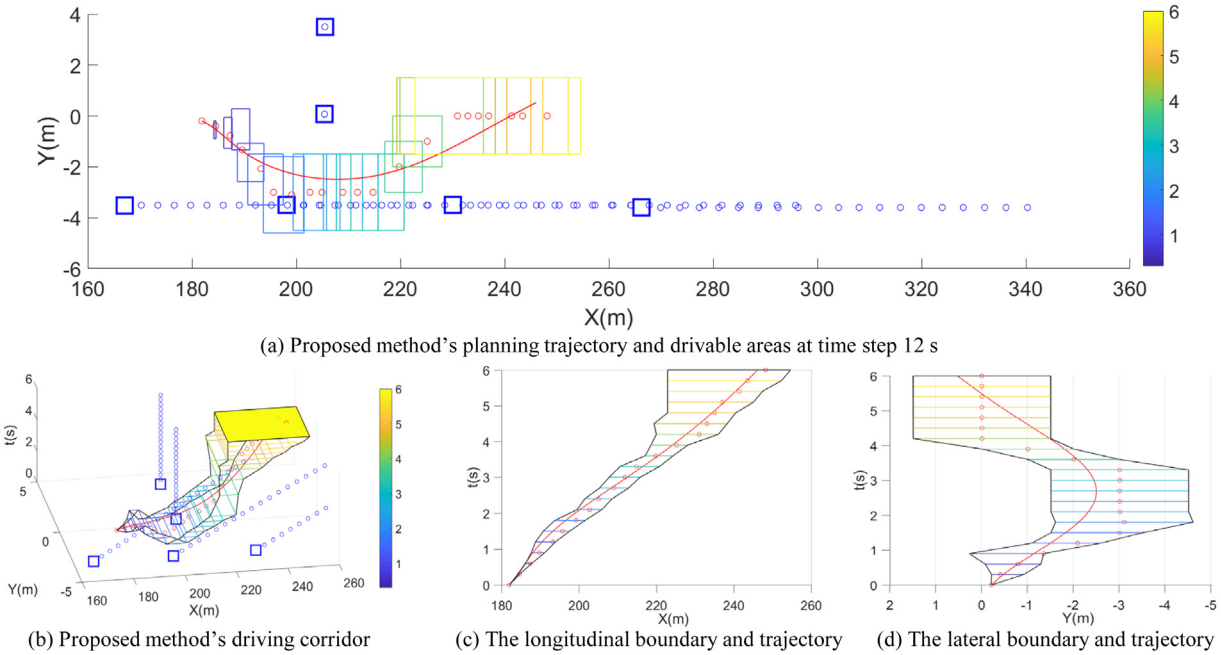
We also compared the proposed method with the spatio-temporal planning algorithms based on the hierarchical optimization (Park et al., 2015). Both algorithms are implemented in Matlab2020a on a computer equipped with an intel i7-9700 3 GHz core and an 16 GB DDR3



(a) Proposed method's planning trajectory and drivable areas at time step 10 s

(b) Proposed method's driving corridor (c) The longitudinal boundary and trajectory (d) The lateral boundary and trajectory

Fig. 15. Proposed method's planning results and drivable areas at time step 10 s, In (a), the rectangle represents the planned driving area at each time, and the color band on the right represents the border color corresponding to the driving area at each preview time, In (b), (c), (d), the red curve represents the last optimized smooth track, The black polyline indicates the boundary of the driving corridor.



(a) Proposed method's planning trajectory and drivable areas at time step 12 s

(b) Proposed method's driving corridor (c) The longitudinal boundary and trajectory (d) The lateral boundary and trajectory

Fig. 16. Driving corridor and the lateral and longitudinal boundaries at time step 12 s, In (a), the rectangle represents the planned driving area at each time, and the color band on the right represents the border color corresponding to the driving area at each preview time, In (b), (c), (d), the red curve represents the last optimized smooth track, The black polyline indicates the boundary of the driving corridor.

memory. As shown in Table 7, our algorithm is slightly faster than the MIQP/MPC algorithm.

Table 7
Comparison of planning time.

	Prediction (ms)	Rough planning (ms)	Fine planning (ms)
Our method	20	860	750
MIQP/MPC	–	1200	2356

7. Conclusion

In this paper, we propose a spatio-temporal trajectory planning method considering probability risk. In the risk assessment model, the uncertainty of vehicle's predicted position and collision risk can be accurately characterized by considering the vehicle dynamic property.

In trajectory planning, we adopt the state sets as the nodes of spatio-temporal multi-fork tree, which can effectively reduce the number of nodes. Experiments show that compared with the traditional method RS+QP and MOBIL+IDM, the average speed per 1000 s is increased by 10.9% and 13.63%. Meanwhile, the number of dangerous behaviors and the acceleration RMS are reduced by 46% and 40.2% than RS+QP, 350% and 41.1% than MOBIL+IDM respectively. Therefore, our proposed method can effectively evaluate the collision risk from other traffic participants in dense traffic flow, and the generalization ability of spatio-temporal planning can enable the vehicle to drive safely in various complex traffic situations.

It is believed that our proposed method can still be improved in relation to human-like planning and computation efficiency. In the future, we plan to apply neural networks to the evaluation of forward reachable sets to speed up the execution of rough planning. In addition, the parallel computing can be adopted for improving real-time performance.

CRediT authorship contribution statement

Xinkang Zhang: Writing – original draft, Methodology, Software, Data curation, Formal analysis. **Bo Yang:** Resources, Supervision, Project administration. **Xiaofei Pei:** Conceptualization, Writing – review & editing, Funding acquisition. **Songxin Lu:** Validation, Formal analysis, Visualization.

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.

Data availability

No data was used for the research described in the article.

Acknowledgments

This work was supported in part by the Foundation of NSFC, China [grant number 52272426]; Laboratory of Hubei Key Advanced Technology for Automotive Components, China[grant number XDQCKF2021009]. (Corresponding author: Xiaofei Pei.)

References

Ajanovic, Z., Lacevic, B., Shyrokau, B., Stolz, M., Horn, M., 2018. Search-based optimal motion planning for automated driving. In: 25th IEEE/RSJ International Conference on Intelligent Robots and Systems. IROS, Madrid, SPAIN, pp. 4523–4530.

Altche, F., de La Fortelle, A., Ieee, 2017. An LSTM network for highway trajectory prediction. In: 20th IEEE International Conference on Intelligent Transportation Systems. ITSC, Yokohama, JAPAN.

Artunedo, A., Villagra, J., Godoy, J., 2022. Jerk-limited time-optimal speed planning for arbitrary paths. Ieee Trans. Intell. Transp. Syst. 23 (7), 8194–8208.

Candela, E., Feng, Y., Mead, D., Demiris, Y., Angeloudis, P., 2021. Fast collision prediction for autonomous vehicles using a stochastic dynamics model. In: 2021 IEEE International Intelligent Transportation Systems Conference, ITSC 2021, September 19, 2021 - September 22, 2021. Institute of Electrical and Electronics Engineers Inc, Indianapolis, IN, United states, pp. 211–216.

Ge, J.I., Schurmann, B., Murray, R.M., Althoff, M., Ieee, 2019. Risk-aware motion planning for automated vehicle among human-driven cars. In: American Control Conference. ACC, Philadelphia, PA, pp. 3987–3993.

Guo, Y.Q., Xu, H.L., Yao, D.Y., Li, L., Ieee, 2020. A real-time optimization-based trajectory planning method in dynamic and uncertain environments. In: 23rd IEEE International Conference on Intelligent Transportation Systems. ITSC, Electr Network.

Hruschka, C.M., Schmidt, M., Topfer, D., Zug, S., Ieee, 2019. Uncertainty-adaptive, risk based motion planning in automated driving. In: IEEE International Conference on Vehicular Electronics and Safety. ICVES, Cairo, EGYPT.

Hu, X.M., Chen, L., Tang, B., Cao, D.P., He, H.B., 2018. Dynamic path planning for autonomous driving on various roads with avoidance of static and moving obstacles. Mech. Syst. Signal Process. 100, 482–500.

Jia, C., Huang, M., Sui, L., 2020. An autonomous vehicle motion planning method based on dynamic programming. In: 17th International Computer Conference on Wavelet Active Media Technology and Information Processing, ICCWAMTIP 2020, December 18, 2020 - December 20, 2020. Institute of Electrical and Electronics Engineers Inc, Chengdu, China, pp. 394–398.

Kalatian, A., Farooq, B., 2022. A context-aware pedestrian trajectory prediction framework for automated vehicles. Transp. Res. C 134.

Karaman, S., Frazzoli, E., 2011. Sampling-Based Algorithms for Optimal Motion Planning, seventh ed. SAGE Publications Inc, pp. 846–894.

Katrakazas, C., Quddus, M., Chen, W.H., Deka, L., 2015. Real-time motion planning methods for autonomous on-road driving: State-of-the-art and future research directions. Transp. Res. C 60, 416–442.

Kesting, A., Treiber, M., Helbing, D., 2007. General lane-changing model MOBIL for car-following models. Transp. Res. Rec.: J. Transp. Res. Board 1999 (1), 86–94.

Khaitan, S., Lin, Q., Dolan, J.M., 2021. Safe planning and control under uncertainty for self-driving. Ieee Trans. Veh. Technol. 70 (10), 9826–9837.

Kim, H., Kim, G., Park, J., Min, K., Kim, D., Huh, K., Ieee, 2019. Action conditioned response prediction with uncertainty for automated vehicles. In: International Symposium on Intelligent Signal Processing and Communication Systems. LSPACS, Taipei, TAIWAN.

Li, P., Pei, X., Chen, Z., Zhou, X., Xu, J., 2022. Human-like motion planning of autonomous vehicle based on probabilistic trajectory prediction. Appl. Soft Comput. 118.

Liu, J.B., Mao, X.Y., Fang, Y.Q., Zhu, D.L., Meng, M.Q.H., Ieee, 2021. A survey on deep-learning approaches for vehicle trajectory prediction in autonomous driving. In: IEEE International Conference on Robotics and Biomimetics (IEEE ROBIO). Sanya, PEOPLES R CHINA, pp. 978–985.

Liu, Z., Pei, X., Chen, Z., Yang, B., Guo, X., 2019. Differential speed steering control for four-wheel distributed electric vehicle. In: ed., April (Ed.), SAE World Congress Experience, WCX 2019, April 9, 2019 - April 11, 2019. SAE International, Detroit, MI, United states.

Manzinger, S., Pek, C., Althoff, M., 2021. Using reachable sets for trajectory planning of automated vehicles. Ieee Trans. Intell. Veh. 6 (2), 232–248.

McNaughton, M., Urmson, C., Dolan, J.M., Lee, J.W., Ieee, 2011. Motion planning for autonomous driving with a conformal spatiotemporal lattice. In: IEEE International Conference on Robotics and Automation. ICRA, Shanghai, PEOPLES R CHINA.

Mu, C., Du, L.L., Zhao, X.M., 2021. Event triggered rolling horizon based systematical trajectory planning for merging platoons at mainline-ramp intersection. Transp. Res. C 125.

Nyberg, T., Pek, C., Col, L.Dal., Noren, C., Tumova, J., Ieee, 2021. Risk-aware motion planning for autonomous vehicles with safety specifications. In: 32nd IEEE Intelligent Vehicles Symposium (IV), Electr Network. pp. 1016–1023.

Park, J., Karumanchi, S., Iagnemma, K., 2015. Homotopy-based divide-and-conquer strategy for optimal trajectory planning via mixed-integer programming. Ieee Trans. Robot. 31 (5), 1101–1115.

Petereit, J., Emter, T., Frey, C.W., Kopfstedt, T., Beutel, A., 2012. Application of hybrid a to an autonomous mobile robot for path planning in unstructured outdoor environments. In: 7th German Conference on Robotics, ROBOTIK 2012, May 21, 2012 - May 22, 2012. VDE Verlag GmbH, Munich, Germany, pp. 227–232.

Qian, L.L., Xu, X., Zeng, Y.J., Li, X.H., Sun, Z.P., Song, H., 2022. Synchronous maneuver searching and trajectory planning for autonomous vehicles in dynamic traffic environments. Ieee Intell. Transp. Syst. Mag. 14 (1), 57–73.

Rocamora, B.M., Pereira, G.A.S., 2022. Parallel sensor-space lattice planner for real-time obstacle avoidance. Sensors 22 (13).

Saini, R., Kale, J.G., Karle, M., Karle, U., 2021. A unique approach for motion planning for autonomous vehicle using modified lattice planner. In: SAE 2021 17th Symposium on International Automotive Technology, SIAT 2021, September 29, 2021 - October 1, 2021, 2021 ed. SAE International, Virtual, Online, India.

Sharma, O., Sahoo, N.C., Puhan, N.B., 2021. Recent advances in motion and behavior planning techniques for software architecture of autonomous vehicles: A state-of-the-art survey. Eng. Appl. Artif. Intell. 101.

Sontges, S., Althoff, M., 2018. Computing the drivable area of autonomous road vehicles in dynamic road scenes. Ieee Trans. Intell. Transp. Syst. 19 (6), 1855–1866.

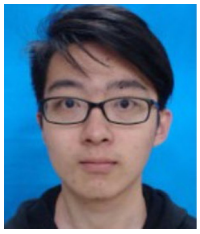
Szepe, T., Assal, S.F.M., 2012. Pure pursuit trajectory tracking approach: Comparison and experimental validation. Int. J. Robot. Autom. 27 (4), 355–363.

Treiber, M., Hennecke, A., Helbing, D., 2000. Congested traffic states in empirical observations and microscopic simulations. Phys. Rev. E 62 (2 A), 1805–1824.

Wei, Y., Avci, C., Liu, J., Belezamo, B., Aydin, N., Li, P., Zhou, X., 2017. Dynamic programming-based multi-vehicle longitudinal trajectory optimization with simplified car following models. Transp. Res. B 106, 102–129.

Xin, L., Kong, Y.T., Li, S., Chen, J.Y., Guan, Y., Tomizuka, M., Cheng, B., 2021. Enable faster and smoother spatio-temporal trajectory planning for autonomous vehicles in constrained dynamic environment. Proc. Inst. Mech. Eng. D-J. Automob. Eng. 235 (4), 1101–1112.

Zhang, T., Fu, M., Song, W., Yang, Y., Wang, M., 2022. Trajectory planning based on spatio-temporal map with collision avoidance guaranteed by safety strip. IEEE Trans. Intell. Transp. Syst. 23 (2), 1030–1043.



Xinkang Zhang received his B.S. in Wuhan University of Technology, China, in 2020. He is now pursuing M.S degree in Vehicle Engineering from Wuhan University of Technology. His research interest includes trajectory planning and decision-making for the autonomous vehicle.

Bo Yang received B.E. degree from Tsinghua University, China, in 1997, and received Ph.D. degree in M.E. from Huazhong University of Science and Technology, China, in 2004. He is now an associate professor in Wuhan University of Technology from 2006. His research field are heavy vehicle design, vehicle dynamics.

Xiaofei Pei was born in Wuhan, Hubei, China in 1985. He received the B.S. degree in Mechanical Engineering from Wuhan University of Technology, Wuhan, China in 2007 and Ph.D. degree in Mechanical Engineering from Beijing Institute of Technology, Beijing, China in 2013. From 2011 to 2012, he was a visiting student with Rutgers University, New Brunswick, NJ, USA. He is now an associate professor in Wuhan University of Technology from 2016. His research interests include vehicle active safety, autonomous vehicle.

Songxin Lu received his B.S. degree from Wuhan University of Technology in China in 2021. He is currently pursuing a M.S. degree in Vehicle Engineering from Wuhan University of Technology. His research interests includes prediction and decision making for the autonomous vehicle.