Multi-Vehicle Collaborative Trajectory Planning in Unstructured Conflict Areas Based on V-Hybrid A*

Biao Xu, *Member, IEEE*, Guan Wang, Zeyu Yang, Yougang Bian, *Member, IEEE*, Xiaowei Wang, and Manjiang Hu

***Abstract*— This work deals with the Multi-Vehicle Trajectory Planning (MVTP) problem in unstructured conflict areas. Compared with MVTP on structured roads, the more complex intersection and the more conflicting motions of vehicles, set higher requirements for the real-time of the planner. To address the issue, this work proposes a centralized decision-making distributed planning framework, to generate trajectories for multiple vehicles navigating conflict areas on unstructured roads. In the multi-vehicle collaboration process, vehicles are assigned a priority upon entering the control area, with higher-priority vehicles treated as dynamic obstacles to be avoided. The distributed planning is divided into three stages: 1) trajectory search utilizing Velocity-Hybrid A* (V-Hybrid A*), 2) path optimization with respect to both dynamic and static obstacles, and 3) speed optimization leveraging the convex space created by the initial solution. In the trajectory search of individual vehicles, each expansion node of V-Hybrid A* is given an acceleration to get a coarse trajectory with discrete velocity. Moreover, a box constraint is established to optimize the path of the coarse trajectory. Then, according to the homotopy class provided by the coarse trajectory, speed optimization is carried out to obtain the final single vehicle trajectory. In our simulation experiments, we randomized the generation of numerous vehicles assigned with varied tasks to evaluate the algorithm's effectiveness. Ultimately, real-vehicle experiments substantiated the algorithm's practical feasibility. The results indicate that the proposed method can significantly improve the traffic efficiency in conflict areas and has practical applicability.**

***Index Terms*— Autonomous vehicles, trajectory planning, trajectory optimization, multi-vehicle collaboration.**

Manuscript received 15 May 2023; revised 7 January 2024 and 27 March 2024; accepted 28 March 2024. This work was supported in part by the National Key Research and Development Program of China under Grant 2021YFB2501800; in part by the National Natural Science Foundation of China under Grant 52102394, Grant 52202493, and Grant 52172384; in part by Hunan Provincial NSF of China under Grant 2023JJ10008; and in part by the Science and Technology Innovation Program of Hunan Province under Grant 2022RC1033. The Associate Editor for this article was Z. Xiao. *(Corresponding author: Zeyu Yang.)*

Biao Xu, Zeyu Yang, Yougang Bian, Xiaowei Wang, and Manjiang Hu are with the State Key Laboratory of Advanced Design and Manufacturing Technology for Vehicle, College of Mechanical and Vehicle Engineering, Hunan University, Changsha 410082, China, and also with Wuxi Intelligent Control Research Institute of Hunan University, Wuxi 214115, China (e-mail: xubiao@hnu.edu.cn; yangzeyu@wicri.org; byg10@foxmail.com; wxw9@163.com; manjiang_h@hnu.edu.cn).

Guan Wang is with the State Key Laboratory of Advanced Design and Manufacturing for Vehicle Body, College of Mechanical and Vehicle Engineering, Hunan University, Changsha 410082, China (e-mail: 178347537@qq.com).

Digital Object Identifier 10.1109/TITS.2024.3383825

I. Introduction

MULTI-VEHICLE trajectory planning has been extensively studied on structured roads (ramp confluence, intersection, three-way intersection, etc.) and has formed relatively mature traffic strategies [1], [2], [3], [4], [5], [6]. However, the challenge intensifies in unstructured conflict areas. As illustrated in Fig. 1, unstructured conflict areas are characterized by irregular boundaries, internal obstacles, and multiple forks at intersections, presenting significant challenges to multi-vehicle trajectory planning. The recent surge in autonomous driving on unstructured roads, including mining sites and parking lots, has significantly heightened the demand for multi-vehicle trajectory planning in these scenarios. While there have been several studies on trajectory planning in unstructured scenarios, the majority have not sufficiently addressed real-time considerations. This study aims to efficiently generate trajectories for multiple vehicles in areas fraught with conflict.

Multi-vehicle trajectory planning can be classified into two categories based on the collaborative methods employed: centralized and distributed. In the centralized approach, the Roadside Management Unit (RMU) serves as the system's nucleus. It collects information about all vehicles within its communication range, formulates instructions in accordance with pre-established collaborative strategies, and dispatches these instructions to the vehicles. This approach allows for optimal coordination among vehicles but struggles with flexibility in the face of unforeseen obstacles such as pedestrians, bicycles, and parked cars [7]. Conversely, the distributed methodology functions independently of roadside infrastructure. Here, vehicles leverage sensing and communication devices to gather data about their immediate surroundings [8]. This approach provides a more adaptable response to obstacles but may lead to reduced vehicular interaction. When the vehicle count scales up, the risk of deadlocks escalates.

Authors in [9], [10], and [11] adopted a centralized decision-making and distributed planning method. Roadside units and vehicles are responsible for decision-making and planning, respectively. The roadside unit compiles traffic decisions and communicates them to vehicles, which then generate trajectories based on these centralized decisions. Since decisions are informed by a global perspective and planning is executed at the vehicle level, this method facilitates the

Fig. 1. Intersection of open pit mine.

resolution of inter-vehicle conflicts and enhances the system's capability to navigate obstacles.

One major category is using the centralized method to allocate traffic resources to vehicles [12]. Generally, vehicles are treated as a unified system for which optimal spatio-temporal resources are then allocated. Hult et al. [13] reconceptualized this allocation task as a mixed integer programming (MIP) problem, introducing a computational algorithm of low complexity that concurrently resolves central coordination and multiple local optimal control issues. Additionally, alternative formulations of the problem have transformed it into a Mixed-integer linear program (MILP) [14], mixed-integer quadratic program (MIQP) [15], and nonlinear program (NLP) [16]. This coupling method can fully exploit the potential for collaboration between vehicles, but when the number of vehicles is too large, this method may fail because the dimensions of collision avoidance constraints will rapidly increase as the number of vehicles increases. Li et al. [17] proposed the adaptive-scaling constrained optimization (ASCO) method, which reduces the dimensionality of collision avoidance constraints by activating them only within a designated ring-like risk area. Nonetheless, ASCO can experience computational delays during high-conflict scenarios where a single risk area is shared by all vehicles. Approaches based on polynomial functions are generally recognized for their expedited computation [18], [19], [20]. But it is not very suitable for unstructured roads. Furthering this avenue of research, Li et al. [21] explored the cooperative operations of automated guided vehicles, developing an offline path library with preset speeds to circumvent collisions at conflict nodes.

A fully distributed strategy employs a reaction-based approach, delegating the task of planning to intelligent vehicles themselves. In this framework, each vehicle dynamically plans its immediate local trajectory in response to the behaviors of surrounding vehicles. Prominent among reaction-based strategies are the Optimal Reciprocal Collision Avoidance (ORCA) method, Model Predictive Control (MPC)-based methods [22], [23], and Potential Field-based adaptive controllers [24], [25]. Li et al. [26] introduced an online trajectory planning method for collision avoidance aimed at enhancing vehicular safety and comfort concurrently. This method leverages a quadratic programming solver, making it computationally viable for online implementation. Wang et al. [27] developed a novel Safety Constraint Interpretive Method (SCIM), which mitigates unnecessary constraints by transforming the safety constraint, thus streamlining the decision-making process. These techniques effectively navigate unforeseen circumstances and alleviate the computational load on the system. However, since vehicles exclusively utilize local information to formulate traffic sequences and control directives, they fall short of achieving global conflict decoupling and optimization. As a result, there are inherent challenges in ensuring overall traffic safety and efficiency.

The centralized decision-making and distributed planning strategy often decomposes the problem into multiple subproblems, managed by backend scheduling or RMU. These strategies typically involve prioritization or grouping methods. The prioritization-based approach sequentially plans the trajectories of each vehicle according to a predetermined priority order. Here, vehicles of lower priority consider those of higher priority as dynamic obstacles [28], [29]. However, illogical rules for priority allocation might lead to planning failures, as the traffic resources for low-priority vehicles could be monopolized by those with higher priority. To resolve this problem, reference [30] introduces a prioritization rule computable online independently by each agent. This rule lowers the priority of agents with multiple homology classes of trajectories when low-priority agents lack sufficient avoidance space, thereby, freeing up more space for maneuvering. Nonetheless, this adjustment may lead to unstable decisions, characterized by sudden changes in curvature or frequent velocity modifications. On the other hand, the grouping-based method identifies vehicles with gaps below a specific threshold as individual sub-problems. It proposes dividing certain areas to facilitate the computation of collaborative trajectories and then integrating these sub-collaborative trajectories [31], [32]. However, forming subgroups becomes challenging in unstructured conflict areas with strong vehicular trajectory conflicts.

From the analysis above, it is evident that most existing planners for unstructured roads do not sufficiently account for the dynamic flow of vehicles. They typically assume a fixed number of vehicles, neglecting scenarios involving the entry of new vehicles. To swiftly address challenges associated with multi-vehicle scenarios, a centralized decision-making and distributed planning framework was adopted to enhance real-time performance. Through this framework, the intelligent scheduling system assigns priority sequences and provides trajectories of high-priority vehicles to new entrants. These vehicles then generate their final trajectories based on the guidance from the scheduling system combined with real-time environmental perceptions. Subsequently, they relay their trajectories back to the intelligent scheduling system. Specifically, the main contributions of our research are:

- A priority-based, centralized decision-making and distributed planning framework is established for managing conflicts in unstructured road areas. This framework aims to leverage the collaborative potential of vehicles and enhance their adaptability to dynamic environments.
- A four-dimensional V-Hybrid A* trajectory search method is proposed, designed to quickly generate initial trajectory guesses and effectively resolve conflicts between vehicles.

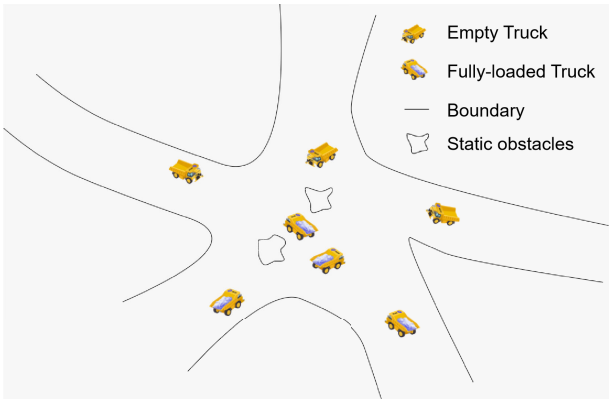


Fig. 2. Top view of intersection conflict area.

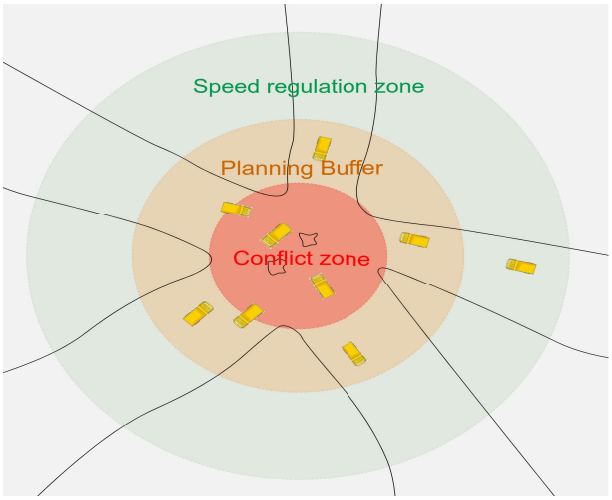


Fig. 3. Different partitions in multi-vehicle collaborative planning.

- A trajectory optimization method is introduced, which accounts for collision avoidance with both dynamic and static obstacles. This method utilizes an ST (Space-Time) diagram, constructed based on the vehicle's actual geometric shape, to ensure the trajectory's smoothness and the comfort of its occupants.

The remainder of this study is organized as follows. Section II introduces the intersection conflict zone model and outlines the framework for centralized decision-making and distributed planning. Section III details a trajectory search method for generating initial guesses. Section IV elaborates on a trajectory optimization method that considers both dynamic and static obstacles. In Section V, we present simulation and real-vehicle experiments conducted to verify the effectiveness of the proposed algorithm. Finally, Section VI provides a summary.

II. Problem Statement and Framework

A. Multi-Vehicle Trajectory Collaborative Planning Problem

The shape of unstructured conflict areas is often irregular, and the intersections within these areas can be complex, leading to an increased number of potential collision points along the driving trajectories of vehicles. Taking the open-pit mining area depicted in Fig. 2 as an example, fully loaded and unloaded mining trucks performing different tasks converge at the intersection conflict area. The complexity of these intersections significantly intensifies trajectory conflicts among the mining trucks. During the process of multi-vehicle collaborative planning, it is paramount not only to consider the impact of irregular obstacles but also to resolve conflicts between vehicles. In addition, compared to general unstructured road multi-vehicle planning problems, this scenario is relatively special because vehicles will continuously enter intersection, which requires a high real-time performance of the planner. This is unbounded in terms of time dimension for the entire system, so it is actually very difficult to seek the global optimal solution using a coupling method.

B. Collaborative Framework

In our system, depicted in Fig. 3, vehicles entering the speed adjustment zone are prioritized by the intelligent scheduling system based on their arrival times, with earlier arrivals granted higher priority. Within this zone, vehicles synchronize their speeds to the lower of the preceding vehicle's speed or the road's speed limit. Furthermore, vehicles maintain a safe following distance through an onboard planning algorithm. The planning buffer's establishment ensures that vehicles can safely stop before reaching the conflict zone in emergencies. In the event that the leading vehicle's planning fails, resulting in it being stationary, trailing vehicles will enter a similar parked and waiting state. Once the leading vehicle resumes motion after successfully implementing its plan, trailing vehicles will adjust their speed according to the ST diagram, based on the lead vehicle's departure time. This planning strategy ensures that each vehicle enters the conflict zone at the maximum safe speed, thereby avoiding collisions.

Overall, the planning algorithm presented in this study can consistently find a solution. In instances where planning is unsuccessful, the algorithm automatically reduces the vehicle's speed within the speed adjustment zone, which alters the timing of its entry into the conflict area. Consequently, the position of dynamic obstacles at the intersection will change, thereby creating adequate space to search for an trajectory.

Fig. 4 illustrates the centralized decision-making and distributed planning framework that we have implemented. This framework primarily encompasses communication between the intelligent scheduling systems and intelligent connected vehicles (ICVs), as well as the vehicles' own trajectory planning processes. The intelligent scheduling system leverages wireless communication devices to gather necessary information and generates traffic decisions based on specific decision-making logic. Meanwhile, the vehicle terminal module, equipped with onboard communication devices, acquires these decision-making results along with other vehicular driving status data. Subsequently, it formulates a final trajectory using a planning algorithm. The vehicle terminal module is composed of four key components: perception, positioning, planning, and control. Upon receiving obstacle data from the perception module and vehicle status information from the positioning module, the planning module devises a smooth trajectory. Following this, the control module executes motion control in accordance with this trajectory to fulfill the desired

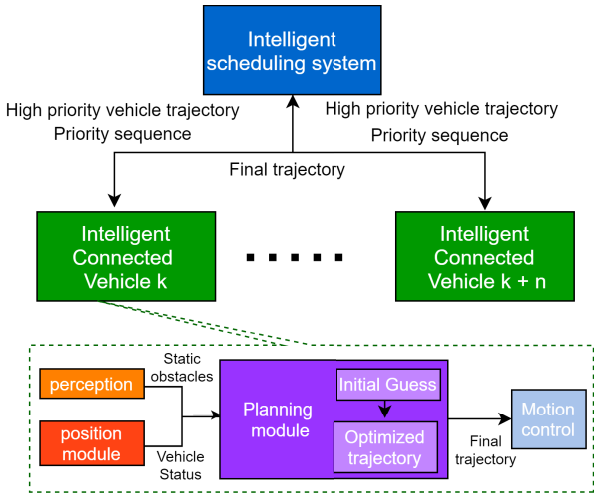


Fig. 4. Framework of centralized decision-making and distributed planning system.

maneuver. Within the planning module, an initial coarse trajectory is created that addresses collision avoidance with both dynamic and static obstacles, albeit without considering the vehicle's kinematic constraints. Furthermore, employing discrete velocities does not ensure safety in a continuous space-time domain. As a remedy, path optimization and speed optimization are conducted based on initial guesses.

This paper uses sequential programming (SP) [33], which is based on the centralized decision-making and distributed planning framework. Within this framework, the intelligent scheduling system assigns planning priorities to incoming vehicles. A vehicle's planning process is initiated only after the trajectories of all vehicles with higher priority have been determined. For individual vehicle planning, all vehicles with higher priority are treated as dynamic obstacles relative to the ego vehicle. In instances where a vehicle halts due to control module errors, malfunctions, or unforeseen reasons, it is reassigned the lowest priority. Subsequently, vehicles influenced by this disruption will also come to a stop and be sequentially assigned lower priorities. These halted vehicles will proceed to re-plan their trajectories following the order of their revised priorities. This mechanism ensures that, as soon as free resource blocks become available, the vehicles that were previously stopped can initiate their movement.

III. V-Hybrid A* Trajectory Search Method

The objective of the trajectory planning task is to identify a safe and collision-free trajectory $\mathcal{T}$ within the spatio-temporal map $\mathcal{M}$. Utilizing the graph search method, the process involves identifying a sequence of nodes from the starting point s_0 to the goal point s_G. Each node $s \in \mathcal{M}$ is defined as the center of a 4-D rectanguloid cell within the world coordinate system, denoted as $s(x, y, \theta, t)$. When the search space is uniformly rasterized, each node s directly corresponds to an individual cell in $\mathcal{M}$.

A. Spatio-Temporal Map

In scenarios with dynamic obstacles, to derive a feasible trajectory, it is imperative to accurately determine the positions of each dynamic obstacle along the time axis. However, the conventional grid maps are inherently two-dimensional and lack temporal resolution. To bridge this gap, we propose the construction of a spatio-temporal map. The fundamental concept involves elongating the trajectory of the vehicle along the time axis at a specified resolution, thereby chronicling the vehicle's trajectory over a given time period. Within this framework, the space and time at an intersection are conceptualized as three-dimensional coordinates encompassing the X, Y, and T dimensions. The three-dimensional space is partitioned into a multitude of resource blocks, each with dimensions dx in length, dy in width and dt in the temporal dimension. Each resource block is thus defined by

$$\mathbb{R}^{\mathrm{XYT}}_{i_x,i_y,i_t} = \begin{cases} & (i_x-1)\cdot dx \le x < i_x \cdot dx \\ (x,y,t) & (i_y-1)\cdot dy \le y < i_y \cdot dy \\ & (i_t-1)\cdot dt \le t < i_t \cdot dt \end{cases} \tag{1}$$

where (i_x, i_y, i_t) denotes the index of each resource block. Multi-vehicle trajectory collaborative planning is fundamentally concerned with the rational allocation of these resource blocks to various vehicles [10]. Fig. 5 illustrates the spatio-temporal map of the intersection's conflict zone. The colored cubes represent the resource blocks occupied by static obstacles and the vehicles at successive time intervals. It is apparent that static obstacles create columns extending vertically along the time axis, whereas dynamic vehicles generate step-like channels, which evolve over time. The stratified structure of the time layer significantly aids the process of trajectory planning and enhances dynamic collision detection.

B. Trajectory Search Algorithm

The basic Hybrid A* algorithm is primarily designed for pathfinding and has limitations in addressing dynamic obstacles. During the trajectory search process, it effectively seeks resource blocks (x_n, y_n, t_n) on the spatio-temporal map that are unoccupied by both dynamic and static obstacles. This implies that the allocation of resource blocks for any two vehicles must not intersect, i.e.,

$$\mathbb{R}^{\mathrm{XYT}}_{i_1,V_1} \cap \mathbb{R}^{\mathrm{XYT}}_{i_2,V_2} = \emptyset \tag{2}$$

where $\mathbb{R}^{\mathrm{XYT}}_{i,V}$ is the vehicle's set of resource blocks.

This study enhances the Hybrid A* algorithm to accommodate the 3-D spatio-temporal map outlined above, effectively addressing road resource conflicts among vehicles and furnishing a robust initial estimate for trajectory optimization, as detailed in Section IV. Recognizing that different heading angles upon reaching the same resource block constitute distinct expansion states, the vehicle's state is represented as the horizontal and vertical coordinates (x, y), heading angle θ, and velocity v, denoted as $s_t(x_t, y_t, \theta_t, v_t)$.

In the preceding section, we lowered the speed within speed adjustment zones, where a bicycle model suffices to depict vehicle mobility [34]. The formulation of expansion states between nodes during the search process is as follows:

$$v_{n+1} = v_n + a_i \cdot \Delta T \tag{3}$$

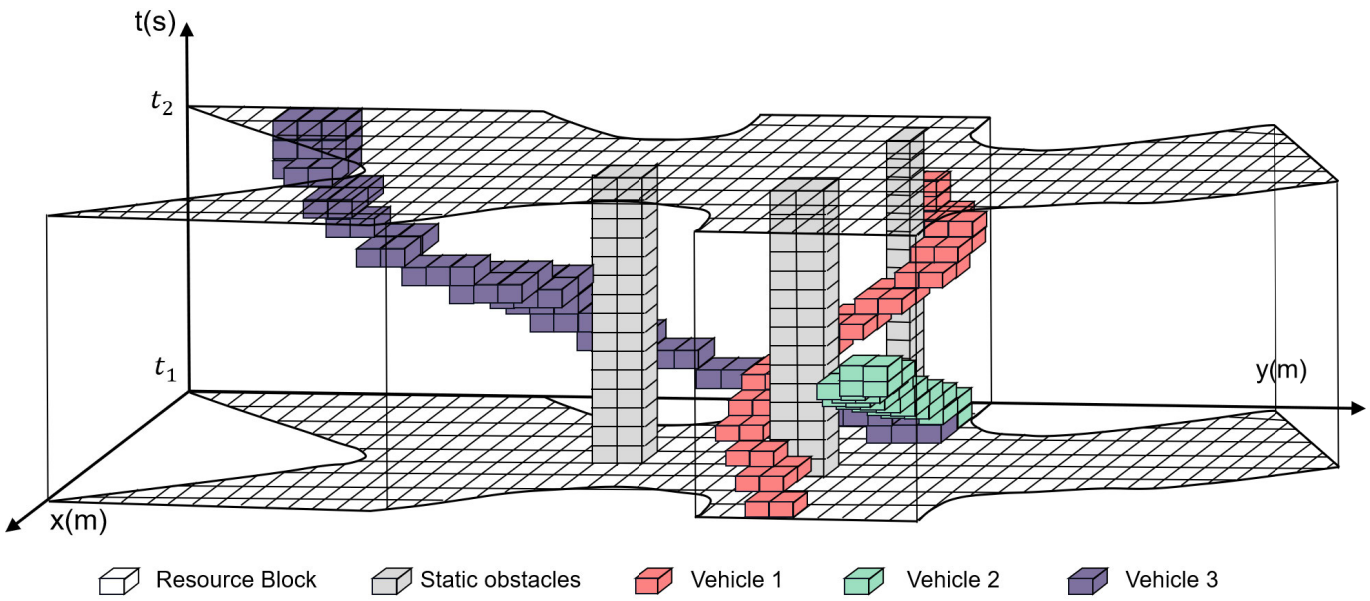


Fig. 5. Spatio-temporal map. The cube represents resource blocks in the spatio-temporal map, the brightly colored one denotes resource blocks occupied by vehicles, the gray one denotes resource blocks occupied by static obstacles, and vacant blocks are not displayed in figure.

$$R_r = \frac{\mathrm{L}}{\tan\delta_j} \tag{4}$$

$$\Delta d = v_{n+1} \cdot \Delta T \tag{5}$$

$$\Delta\theta = (\frac{\Delta d}{\mathrm{L}}) \cdot \tan\delta_j \tag{6}$$

$$\theta_{n+1} = \theta_n + \Delta\theta \tag{7}$$

$$x_{n+1} = x_n + R_r \cdot (\sin\theta_{n+1} - \sin\theta_n) \tag{8}$$

$$y_{n+1} = y_n + R_r \cdot (\cos\theta_n - \cos\theta_{n+1}) \tag{9}$$

$$t_{n+1} = t_n + \Delta T \tag{10}$$

$$a_{min} \le a_i \le a_{max} \tag{11}$$

$$-\delta_{max} \le \delta_j \le \delta_{max} \tag{12}$$

$$0 \le v_{n+1} \le v_{max} \tag{13}$$

where a_i and δ_j represent the sampled accelerations and front wheel angles respectively, with L denoting the vehicle wheelbase, Δd indicating the expansion step length, R_r representing the turning radius, and δ_{max} and δ_{min} denoting the maximum and minimum front wheel angles respectively. ΔT signifies the time resolution. Employing a large ΔT can expedite the search process and smoothen speed changes, however, in complex scenarios, it often leads to search failures. Conversely, a small ΔT exhibits strong adaptability to complex scenes, but compromises trajectory quality and prolongs calculation time. We set maximum accelerations a_{max} and minimum accelerations a_{min} to constrain the speed sampling range during node expansion. The distance Δd between nodes is no longer fixed but determined by the expansion speed, allowing for flexible responses to dynamic obstacles. We opt for a uniform velocity model over a more intricate uniform acceleration model to mitigate computational complexity during each expansion. This choice does not impede exploration of convex space, acceleration results in a smaller expansion length compared to the precise model, yet the vehicle safety distance typically exceeds this length, indicating safety checks have been conducted for this space segment. Conversely, deceleration leads to an extension distance longer than that of the precise model, thus ensuring a safer extension.

During the expansion process, the closed set not only retains nodes selected into the trajectory but also preserves resource blocks occupied by dynamic and static obstacles in the spatio-temporal map inaccessible to the current vehicle. The open set continues to hold candidate nodes awaiting examination. As the time dimension expands, nodes now incorporate speed attributes, there are some improvements in the cost function for node evaluation. The evaluation function $f(\cdot)$ for node-n is expressed as follows:

$$f(n) = g(n) + h(n) \tag{14}$$

$$\begin{aligned} h(n) &= \|s_n - s_G\|_2 \\ &= |x_n - x_G| + |y_n - y_G| + |t_n - t_G| \end{aligned} \tag{15}$$

$$\begin{aligned} g(n+1) = g(n) &+ \Delta d + w_v \cdot |v_{n+1} - v_n| \\ &+ w_{ref} \cdot |v_{n+1} - v_{ref}| + w_\delta \cdot |\theta_{n+1} - \theta_n| \end{aligned} \tag{16}$$

where $g(\cdot)$ and $h(\cdot)$ represent the cumulative past costs and future expected costs respectively, the $g(\cdot)$ function comprises five terms. These include the cost inherited from the previous node, the cost of expanding distance, the cost of speed change, the cost of maintaining reference speed, and the cost of changing direction. Notably, the cost of speed change serves to mitigate frequent speed alterations in the trajectory. However, this may lead to prolonged periods of either low or high speeds, compromising efficiency or safety. Introducing a reference speed maintenance cost effectively addresses this issue, ensuring the vehicle maintains a safe and efficient speed at intersections. Remarkably, when the time to reach the target point is unspecified, $h(\cdot)$ solely reflects the difference in location between the two nodes. Upon defining the arrival time, $h(\cdot)$ incorporates the temporal difference between the nodes, motivating the vehicle to reach the destination at the predefined time.

Algorithm 1 V-Hybrid A* Algorithm for Spatio-Temporal Trajectory Search

Input: spatio-temporal map; start node s_0; end node s_G;
Output: discrete trajectory;

Initialize the trajectory set with the start node;
Initialize the open set with the start node, initialize closed set with resource blocks $s_{obs}(x_n, y_n, t_n)$ occupied by dynamic and static obstacles;
while the open set is not empty **do**
  Take the node current with the lowest cost as the vehicle states s_c;
  Remove s_c from the open set and add it to the closed set;
  if IsFittingSuccess(s_c, s_G) **then**
    TracePath();
    output discrete trajectory;
  else
    FindExpandNode(s_c);
    for each child node in ascending speed order **do**
      if in closed set **then**
        ignore it;
        break;
      else
        if child node is deceleration node **then**
          generate intermediate nodes;
          if intermediate node in closed set **then**
            add child node to the closed set;
            break;
          end if
        end if
        if is not in the open set **then**
          add child node to the open set;
        else
          compare g value and update in openset;
        end if
      end if
    end for
  end if
end while

The pseudocode of the V-Hybrid A* algorithm is presented in Algorithm 1. The IsFittingSuccess(s_c, s_G) function determines whether fitting is successful. If the current node s_c is in proximity to the end node s_G, Dubins curves [35] are employed to fit two nodes, and a reasonable speed is sampled based on the current node speed. Subsequently, the entire time range of the fitted trajectory is calculated, and if no collisions occur, true is returned; otherwise, false is returned. Specifically, multiple velocities are sampled based on the maximum and minimum acceleration limits to generate multiple fitting trajectories. The fitted trajectory without collisions and the highest velocity are selected as the final fitting trajectory.

The TracePath() function is utilized to trace back to the start node s_0 and splice the fitting trajectory to obtain the initial discrete trajectory. The child nodes of the current node s_c are generated through the FindExpandNode(s_c) function. Samples

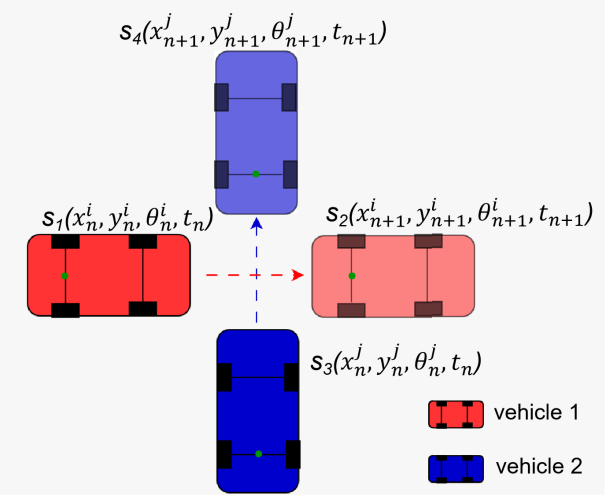


Fig. 6. Possible collisions caused by discrete search times.

are taken for $a > 0$, $a = 0$, and $a < 0$ to obtain acceleration, constant velocity, and deceleration nodes respectively.

Due to the discrete nature of time, the travel distance of the ego vehicle may sometimes exceed the width of another vehicle within a single time step. Taking Fig. 6 as an example, the red ego vehicle transitions from state $s_1(x^i_n, y^i_n, \theta^i_n, t_n)$ to $s_2(x^i_{n+1}, y^i_{n+1}, \theta^i_{n+1}, t_{n+1})$, while the blue obstacle vehicle travels from $s_3(x^j_n, y^j_n, \theta^j_n, t_n)$ to $s_4(x^j_{n+1}, y^j_{n+1}, \theta^j_{n+1}, t_{n+1})$. Although such scenarios may not result in conflicts within a discrete search space, they are undesirable in a continuous space. To address this issue, it is necessary to determine whether the gap between node extensions is traversable.

Due to acceleration limitations, gaps between acceleration, constant velocity, and deceleration nodes of the vehicle can be filled by the vehicle's shape, with the maximum gap typically occurring between the deceleration node and the parent node. Specifically, the gap between acceleration, constant velocity, and deceleration nodes is dictated by the speed difference, while the gap between deceleration nodes and parent nodes is influenced by the speed of the deceleration nodes. Higher vehicle speeds result in larger deceleration node speeds, consequently leading to wider gaps. Therefore, an intermediate detection node is introduced during collision detection with deceleration nodes, extracting sufficiently dense trajectory points of obstacle vehicles for collision detection. As low-speed nodes compensate for gaps between high-speed nodes and parent nodes, successful expansion of low-speed nodes is a prerequisite for the successful expansion of high-speed nodes. If the deceleration node is deemed invalid, further expansion in that direction ceases. The rationality of the fitting trajectory is verified by introducing more intermediate nodes. It is noteworthy that node expansion between nodes does not directly involve parking and waiting nodes [36]; rather, it occurs only at low speeds. We are considering that when the speed is too fast, this node expansion does not conform to the actual deceleration performance of the vehicle, which will lead to the failure of the speed optimization in the next section.

The nodes of the V-Hybrid A* algorithm are expanded according to sampled accelerations. From a macro perspective,

this process resembles simulating acceleration and deceleration based on changes in obstacles, akin to human driving behavior. Consequently, the resulting search trajectory may involve acceleration or deceleration to evade obstacles. This approach enables the derivation of avoidance decisions based on the actual maneuvering space of the vehicle, rather than adhering to rigid rules during conflict resolution between vehicles. Consequently, it reduces the likelihood of deadlock occurrence.

IV. Trajectory Optimization

The trajectory obtained in the previous section lacks smoothness, and the discrete nature of speed not only diminishes ride comfort but also leads to significant tracking errors in the control module. Further optimization of the trajectory is necessary to render it suitable for the control module. The presence of moving obstacles typically introduces non-convex constraints to optimization problems, posing a significant challenge in trajectory optimization. Dealing with dynamic obstacles is particularly challenging due to the uncertainty regarding whether to accelerate or decelerate to avoid them. Notably, the initial guess obtained using V-Hybrid A* includes velocity information, and the method of sampling acceleration largely adheres to the kinematic constraints of the vehicle. Thus, leveraging these initial guesses enables the resolution of dynamic constraint problems. In the subsequent discussion, the trajectory optimization problem is decomposed into two quadratic programming (QP) problems: path optimization and speed optimization.

A. Path Optimization

Path optimization can be formulated as a QP problem consisting of objective functions, endpoint constraints, and security box constraints as follows.

$$min\ \mathcal{F}_p = \omega_s \cdot f_s(\mathcal{X}) + \omega_r \cdot f_r(\mathcal{X}) + \omega_l \cdot f_l(\mathcal{X}) \tag{17}$$

$$\begin{aligned} s.t.\ & \left[x(0), y(0)\right] = \left[\mathrm{x}_0, \mathrm{y}_0\right] \\ & \left[x(\mathrm{N}), y(\mathrm{N})\right] = \left[\mathrm{x}_{\mathrm{end}}, \mathrm{y}_{\mathrm{end}}\right] \\ & \mathrm{xlb}_k \le x(k) \le \mathrm{xub}_k \\ & \mathrm{ylb}_k \le y(k) \le \mathrm{yub}_k \end{aligned} \tag{18}$$

In path optimization, only location information is required, hence only $(x(k), y(k))$ is involved in the formula (17) (18). Within the the cost function $\mathcal{F}_p$, $f_s(\mathcal{X})$ is employed to ensure the smoothness of the path. Specifically, it is expressed as:

$$f_s(\mathcal{X}) = \sum_{k=0}^{N-2} [(x_k + x_{k+2} - 2x_{k+1})^2 + (y_k + y_{k+2} - 2y_{k+1})^2] \tag{19}$$

The second term $f_r(\mathcal{X})$ is deployed to encourage the optimized path to fit the original path, thereby preserving the original homotopy class. Suppose that the initial trajectory is denoted as $\mathrm{X}_{ref}(k)$, $k \in [0,\mathrm{N}]$. $f_r(\mathcal{X})$ is concretely written as

$$f_r(\mathcal{X}) = \left\| \mathcal{X}(k) - \mathrm{X}_{ref}(k) \right\|$$

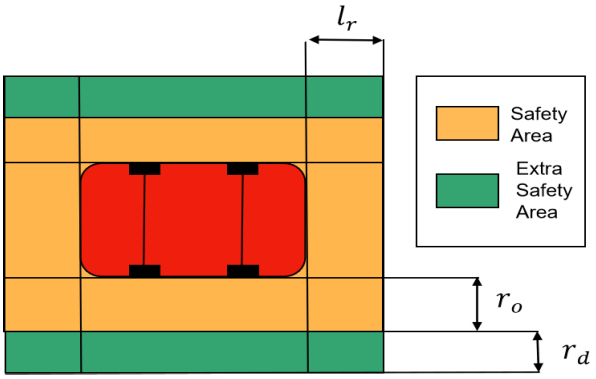


Fig. 7. Safe area around vehicle.

$$= \sum_{k=0}^{N} [(x_k - \mathrm{x}_{ref}(k))^2 + (y_k - \mathrm{y}_{ref}(k))^2] \tag{20}$$

The last term $f_l(\mathcal{X})$ is used to ensure the smoothed path to have a uniform length between waypoints, whose cost is calculated by

$$\begin{aligned} f_l(\mathcal{X}) &= \|\mathcal{X}(k+1) - \mathcal{X}(k)\| \\ &= \sum_{k=0}^{N-1} [(x_{k+1} - x_k)^2 + (y_{k+1} - y_k)^2] \end{aligned} \tag{21}$$

wherein $\omega_s, \omega_r, \omega_l$ is the corresponding weight parameter.

As depicted in Fig. 7, an additional safety space is incorporated to prevent the optimized trajectory from colliding with dynamic obstacles. During trajectory search and path optimization, the actual size of the vehicle is extended to the boundary of the green additional safety area. However, during speed planning, the vehicle only extends to the edge of the yellow safety area. This is because path optimization, when navigating through narrow passageways, may cause waypoints to deviate from the original path, potentially leading to collisions.

In Fig. 8, with two planned vehicles already present, the original path of the current planned vehicle passes through the middle of the two paths. To mitigate spatial conflicts with the optimized path, we limit the degree of waypoint offset based on additional safety distances. Notably, there is no need to introduce an additional safety distance in the longitudinal direction, as collisions resulting from a decrease in longitudinal distance can be averted by adjusting the speed. By assuming equal optimization degrees in the x and y directions, the maximum optimization distance is along the diagonal direction of the rectangular box. Consequently, the maximum degree of path optimization can be calculated as follows:

$$\Delta x_k = \Delta y_k = \frac{r_d}{\sqrt{2}} \tag{22}$$

In addition, it is expected that the change of heading angle will be relatively smooth near the starting point and the ending point. Then a coefficient needs to be added to the variation in the formula (22).

$$\Delta x_k = \Delta y_k = \frac{r_d}{\sqrt{2}} \cdot \frac{1}{1 + e^{(4-\mu)}} \tag{23}$$

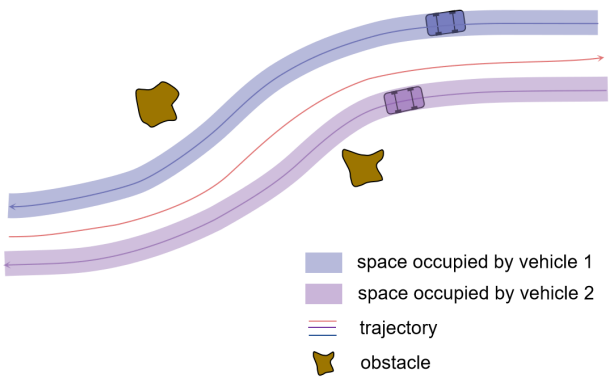


Fig. 8. Partial screenshot of vehicle path.

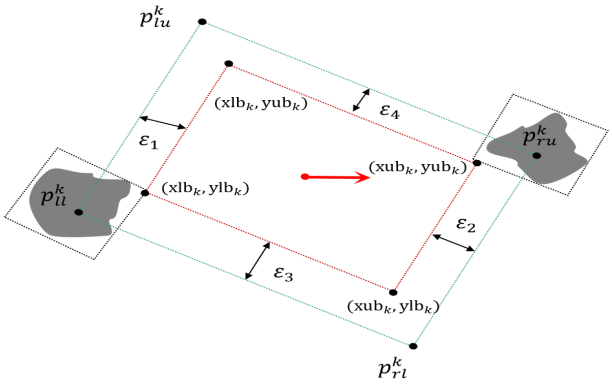


Fig. 9. Generation of box constraints.

$$\mu = \begin{cases} k, k \le \dfrac{N}{2} \\ N-k, k > \dfrac{N}{2} \end{cases} \tag{24}$$

The constraint box is defined based on the vehicle coordinate system, as illustrated in Fig. 9. The green initial constraint box is initialized according to the extra safety distance. Adjusting waypoints within this box ensures collision-free navigation with dynamic obstacles, as the maximum change amplitude of the waypoint has been accounted for when calculating Δx_k and Δy_k. Static obstacle information is obtained in the form of rectangular envelopes through the perception module and is gridified based on the maximum envelope size. We assess whether the grid points inside the constraint box coincide with the grid points of obstacles, adjusting the constraint box to avoid conflicts with any obstacle grid points. For each waypoint, the maximum optimization box can be determined, with its four corner coordinates denoted as $p_{ru}^k(x_k+\Delta x_k, y_k+\Delta y_k)$, $p_{rl}^k(x_k+\Delta x_k, y_k-\Delta y_k)$, $p_{lu}^k(x_k-\Delta x_k, y_k+\Delta y_k)$, and $p_{ll}^k(x_k-\Delta x_k, y_k-\Delta y_k)$. The adjusted constraint box can be represented by the following formula (18)

$$\begin{aligned} \text{xlb}_k &= x_k - \Delta x_k + \varepsilon_1 \\ \text{xub}_k &= x_k + \Delta x_k - \varepsilon_2 \\ \text{ylb}_k &= y_k - \Delta y_k + \varepsilon_3 \\ \text{yub}_k &= y_k + \Delta y_k - \varepsilon_4 \end{aligned} \tag{25}$$

During the path optimization process, the influence of changes in vehicle heading angle on box constraints was not considered, resulting in box constraints that actually exceeded the safety distance set during trajectory search in terms of distance between the front and back of the vehicle. However, there is no need for concern because the heading angle change between points is minimal. Any risk of front and rear collision can be addressed during speed optimization, where only greater deceleration or acceleration of the vehicle is required.

B. Speed Optimization

Based on initial guesses, a convex space for speed planning can be obtained, and speed optimization can also be transformed into a QP problem as follows.

$$min\ \mathcal{F}_v = \omega_v \cdot f_v(\mathcal{S}) + \omega_a \cdot f_a(\mathcal{S}) + \omega_j \cdot f_{jerk}(\mathcal{S}) \tag{26}$$

$$\begin{aligned} s.t.\ & [s(0), \dot{s}(0), \ddot{s}(0)] = [0, \text{v}_{\text{start}}, \text{a}_{\text{start}}] \\ & [s(\text{N}), \dot{s}(\text{N}), \ddot{s}(\text{N})] = [\text{s}_{\text{end}}, \text{v}_{\text{end}}, 0] \\ & s_k + \dot{s}_k \Delta T + \frac{1}{3}\ddot{s}_k \Delta T^2 - s_{k+1} + \frac{1}{6}\ddot{s}_{k+1}\Delta T^2 = 0 \\ & \dot{s}_k + \frac{1}{2}\ddot{s}_k \Delta T - \dot{s}_{k+1} + \frac{1}{2}\ddot{s}_{k+1}\Delta T = 0 \\ & 0 \le \dot{s}(k) \le \text{v}_{\max} \\ & \text{a}_{\min} \le \ddot{s}(k) \le \text{a}_{\max} \\ & 0 \le \dot{s}(k) \le \left(\frac{\text{a}_{\text{ymax}}}{\kappa}\right)^{\frac{1}{2}} \\ & \mathcal{O}_{lb} < s_k < \mathcal{O}_{ub} \end{aligned} \tag{27}$$

where, $\mathcal{S}(s(k), \dot{s}(k), \ddot{s}(k))$ represents the distance, velocity, and acceleration of the optimized path. The first term $f_v(\mathcal{S})$ in the cost function $\mathcal{F}_v$ is used to encourage vehicles to drive at a reference speed, written in detail as follows

$$f_v(\mathcal{S}) = \sum_{k=0}^{N} (\dot{s}_k - \text{v}_{\text{ref}})^2 \tag{28}$$

v_{ref} is the predefined intersection reference speed, which should be less than the road speed limit $\text{v}_{\max}$. Vehicle is expected to have a small acceleration, $f_a(\mathcal{S})$ is designed to drive the vehicle without excessive acceleration and deceleration maneuvers. It is expressed in detail as

$$f_a(\mathcal{S}) = \sum_{k=0}^{N} \ddot{s}_k^2 \tag{29}$$

In addition, acceleration change of the vehicle should not be too drastic. The third item $f_{jerk}(\mathcal{S})$ has been added to ensure that the vehicle has a smaller jerk, specifically defined as

$$f_{jerk}(\mathcal{S}) = \sum_{k=0}^{N-1} (\ddot{s}_{k+1} - \ddot{s}_k)^2 \tag{30}$$

The three coefficients ω_v, ω_a, and ω_j in the cost function represent their respective cost weights. The two-point boundary constraint in formula (27) is associated with the vehicle's speed upon entering the intersection and the terminal speed of the search trajectory, with the expectation of zero acceleration at the intersection exit.

To ensure the optimized trajectory exhibits continuous acceleration and achieves a smooth and comfortable trajectory,

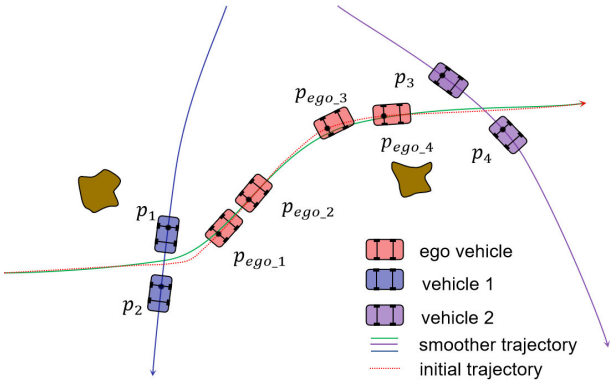


Fig. 10. The process of opening up convex spaces.

equality constraints are imposed on the third and fourth lines of the formula. The finite-term Taylor expansion of s_{k+1} and $\dot{s}_{k+1}$ with respect to time yields the following equation:

$$\begin{aligned} s_{k+1} &= s_k + \dot{s}_k \Delta T + \frac{1}{2}\ddot{s}_k \Delta T^2 \\ &\quad + \frac{1}{6}\frac{\ddot{s}_{k+1} - \ddot{s}_k}{\Delta T}\Delta T^3 + o(\Delta T^3) \\ \dot{s}_{k+1} &= \dot{s}_k + \ddot{s}_k \Delta T + \frac{1}{2}\frac{\ddot{s}_{k+1} - \ddot{s}_k}{\Delta T}\Delta T^2 + o(\Delta T^2) \end{aligned} \tag{31}$$

This ensures that the third derivative jerk $\frac{d^3 s}{dt^3}$ is constant, while the fourth derivative and above are zero. By disregarding the infinitesimal terms $o(\cdot)$ in Equation (31), we obtain the equation constraint that satisfies acceleration continuity in equation (27).

The first three inequality constraints are utilized to confine speed and acceleration within reasonable ranges. In the third inequality constraint, κ denotes the curvature of the path, and a_{ymax} represents the maximum lateral acceleration. The inclusion of a_{ymax} aims to prevent vehicles from traversing curves at high speeds. The final set of constraints pertains to collision avoidance, which ensures vehicles do not collide with other dynamic obstacles. Here, $\mathcal{O}_{lb}$ refers to the lower boundary established by vehicles that need to accelerate to surpass it, while $\mathcal{O}_{ub}$ indicates the upper boundary formed by vehicles that need to decelerate to avoid collision.

Following path optimization, both the length and position of the trajectory undergo changes, affecting the calculated waypoint times derived from waypoint speed. Consequently, this alteration fails to create a convex space for speed planning. Despite the initial trajectory obtained through trajectory search being discrete and non-smooth, it remains collision-free, as it comprises unused resource blocks within the spatio-temporal map. Therefore, while the smoother trajectory serves as the basis for establishing the ST diagram, the initial trajectory opens up the convex space required for speed planning.

The pseudocode for creating convex space ST diagram is presented in Algorithm 2. The function FindConflictPoints($\mathcal{T}_i$, $\mathcal{T}_s$) is employed to identify points that exhibit spatial conflicts between two trajectories $\mathcal{T}_{init}$ and $\mathcal{T}_s$. During function execution, rectangular boxes with a safe distance are generated using the waypoints on the two trajectories as the center points of the vehicle's rear axle. If the two rectangular boxes overlap,

Algorithm 2 Algorithm for Creating Convex Space in ST Diagram

Input: Trajectories of high priority vehicles $\mathcal{T}_{sub}$; initial trajectory $\mathcal{T}_{init}$; smoother trajectory $\mathcal{T}_s$;
Output: Upper and lower boundaries of convex space $\mathcal{O}_{lb}, \mathcal{O}_{ub}$;

```
O_lb, O_ub ← ∅;
for T_i in T_sub do
    P_c ← FindConflictPoints(T_i, T_s);
end for
for P_i in P_c do
    s_pro ← the projected distance of point P_i on initial trajectory T_init;
    s_init ← the distance on the initial trajectory T_init at the time of the point P_i;
    j ← index of point on the smoother trajectory T_s closest to point P_i;
    if s_pro < s_init then
        while Iscollision(P_i, T_s(j)) do
            j = j + 1;
        end while
        O_lb ← T_s(j);
    else
        while Iscollision(P_i, T_s(j)) do
            j = j − 1;
        end while
        O_ub ← T_s(j);
    end if
end for
```

the respective points are considered to be in conflict, and their information is stored. As illustrated in Fig. 10, the trajectory segments from p_1 to p_2 of vehicle 1 and from p_3 to p_4 of vehicle 2 exhibit spatial conflicts with the smoother trajectory $\mathcal{T}_s$. The positions of the ego vehicle on the initial trajectory $\mathcal{T}_{init}$ at corresponding times p_1, p_2, p_3, and p_4 are denoted as p_{ego_1}, p_{ego_2}, p_{ego_3}, and p_{ego_4}, respectively. On the initial trajectory, the distance s_{ego_1} corresponding to p_{ego_1} is greater than the distance s_{pro_1} projected by p_1, whereas the distance s_{ego_4} corresponding to p_{ego_4} is less than the distance s_{pro_4} projected by p_4. This indicates that the avoidance decisions for vehicle 1 and vehicle 2 during trajectory search entail acceleration and deceleration, respectively. Consequently, vehicles requiring acceleration to avoid collision, such as vehicle 1, contribute to the formation of the lower bound $\mathcal{O}_{lb}$ of the convex space, while vehicles necessitating deceleration, such as vehicle 2, contribute to the upper bound $\mathcal{O}_{ub}$ of the convex space. The variable j is utilized to record the index of the first non-collision point. At each moment, the position of the vehicle must not surpass the upper boundary or dip below the lower boundary. Consequently, when searching for the lower boundary, the index j on $\mathcal{T}_s$ must be incremented until the first non-collision point is encountered, with the distance of this point serving as the lower boundary at the corresponding time. Conversely, for the upper boundary, the index j must be decremented.

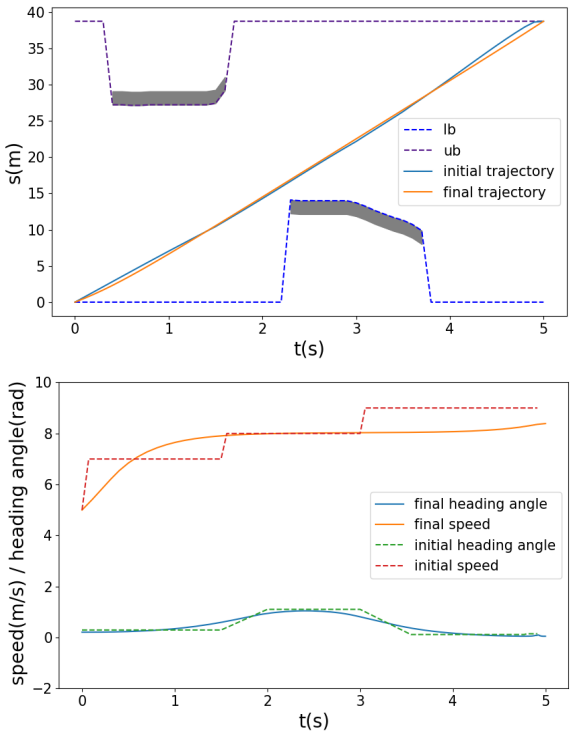


Fig. 11. ST diagram and optimized trajectory in Fig.10.

The ST diagram and optimization curve depicted in Fig. 11 illustrate the scene presented in Fig. 10. The two irregular polygons positioned above and below represent the non-invasive space formed by vehicle 1 and vehicle 2, respectively. The shape of the polygonal edge reflects the speed of the obstacle vehicle along the ego vehicle trajectory. It is evident that search algorithms consistently provide a secure channel in the ST diagram. Despite the initial trajectory velocity being discrete and the velocity curve lacking sufficient smoothness, optimization yields a trajectory that meets the specified requirements. It is noteworthy that in some instances, the smoother trajectory following path optimization intersects irregular polygons in the ST diagram. As mentioned earlier, changes in the position and length of the smoother trajectory result in the abandonment of certain obstacle avoidance decisions made by the initial trajectory. Hence, the convex space is established based on the initial trajectory. Furthermore, the convex space provided by the initial trajectory may not always be precise due to the discrete speeds failing to ensure the continuity of vehicle acceleration and deceleration. This may lead to vehicles adopting extreme avoidance strategies, such as rapid acceleration to overtake one vehicle and rapid deceleration to avoid another vehicle in a constrained space. However, in continuous space, the process of acceleration and deceleration takes longer, rendering such extreme maneuvers impractical. Under such conditions, the safety space between two vehicles diminishes, necessitating the establishment of a reasonable safety distance between the front and back to prevent such extreme search outcomes.

V. Simulation and Experiment

To validate the effectiveness and applicability of the proposed method, simulations were conducted using the Robot Operating System (ROS) to simulate the positioning module, control module, perception module, planning module, and intelligent scheduling system. The parameter settings utilized in the simulation are detailed in Table I. We conducted a comparative analysis of proposed algorithm's performance against rule-based methods under varying traffic flows. Furthermore, to ascertain the practical applicability of the algorithm, a real-vehicle experiment was conducted.

TABLE I
A List of Simulated Parameter Settings

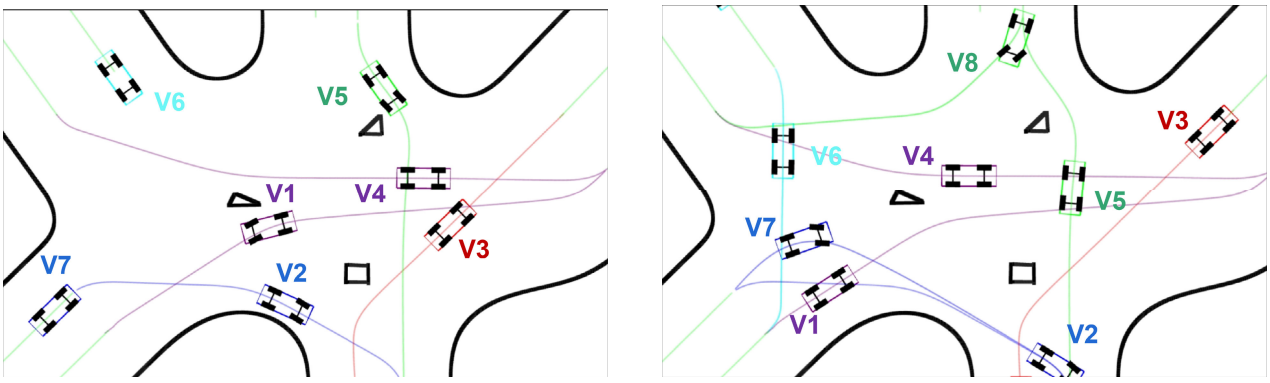
Parameters	Description	Setting
ΔT	unit step of time	0.5 s
a_{max}	maximum acceleration	4.0 m/s^2
a_{min}	minimum acceleration	-6.0 m/s^2
δ_{max}	maximum steer angle	$\frac{\pi}{6}$ rad
a_{ymax}	maximum lateral acceleration	0.2g
v_{max}	maximum velocity	10 m/s
L	wheel base	2.85 m
w_v	speed change coefficient	0.5
w_{ref}	coefficient for deviation from reference speed	0.2
w_δ	direction change coefficient	2.468
ω_s	smoothing coefficient	10^5
ω_r	deviation coefficient of reference line	1.0
ω_l	uniform spacing coefficient	1.0
ω_v	coefficient for deviation from reference speed	50.0
ω_a	acceleration penalty coefficient	100.0
ω_{jerk}	jerk penalty coefficient	10.0

A. Simulation

In the simulation scenario, a five-fork intersection was established, with vehicles entering the intersection randomly at any given time. Fig. 12 illustrates the driving status of vehicles at different times within the conflict zone. The shape of the intersection is highly intricate, containing irregular obstacles. Consequently, trajectory conflicts between vehicles intensify. For instance, vehicle V4 may initially accelerate to avoid vehicle V3, while newly entering vehicle V5 decelerates to avoid vehicle V4. Meanwhile, vehicle V6 may need to accelerate to avoid vehicle V4 and subsequently decelerate to avoid vehicle V7. This intricate coupling relationship underscores the complexity of the scenario, emphasizing the effectiveness of the decoupling method in resolving conflicts step by step.

Furthermore, vehicle V7 and vehicle V2 share the same task. However, vehicle V7 generates a trajectory distinct from vehicle V2 due to the influence of vehicle V1. This observation highlights that when sufficient space is available, vehicles prioritize adjusting their paths to avoid collisions over reducing speed.

In Fig. 13, a speed profile of 16 vehicles at intersections without obstacles is depicted, revealing that most vehicles maintain speeds near the reference speed. Only a small subset of vehicles accelerates or decelerates to avoid collisions. This behavior arises because within obstacle-free intersection areas, most vehicles can avoid collisions by adjusting their path shapes without necessitating speed adjustments. Conversely,



(a) T1 = 8s (b) T2 = 11s

(c) T3 = 12s (d) T4 = 14s

Fig. 12. Case 1: Snapshot of intersection traffic. The polygon in the center of the intersection denote obstacles such as falling rocks that cannot be crossed in the mining area.

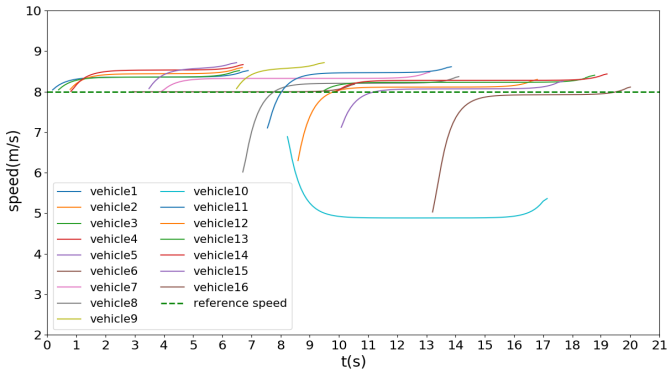


Fig. 13. Velocity profiles of 16 vehicles at an unobstructed intersection.

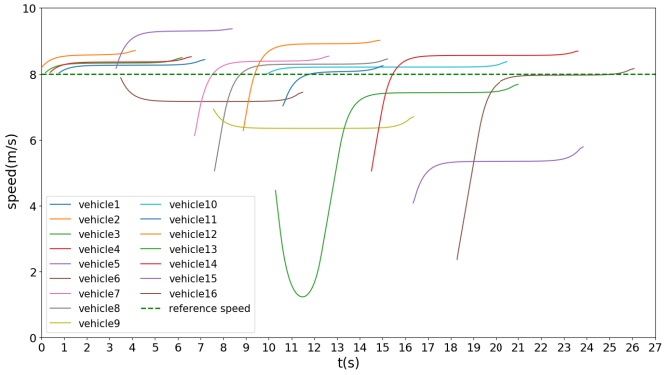


Fig. 14. Velocity profiles of 16 vehicles at intersection with obstacles.

Fig. 14 shows the speed profile for 16 vehicles at intersections with obstacles, illustrating that, despite predominantly maintaining speeds near the reference, the speed deviations and adjustment ranges for individual vehicles are significantly larger.

Moreover, the search algorithm provides different homotopy classes based on the occupancy of spatio-temporal resources. As shown in Fig. 15, vehicles V4 and V1 have the same task of traversing from road 2 to road 4, whereas vehicle V2 is tasked with moving from road 5 to road 2. Due to the space on the left side of obstacle 1 being occupied by vehicle V2, vehicle V4 has a trajectory different from vehicle V1 and their trajectories are not of the same homotopy class. The search algorithm proposed in this study can identify narrow avoidance spaces within the road. For example, vehicle V5, maneuvering from road 3 to road 4, slightly reduces its speed to avoid colliding with vehicle V1, which is traveling from road 2 to road 4. When vehicle V6 subsequently enters the scene, it finds only a narrow avoidance space remaining on the left side of road 4. Despite this, vehicle V6 successfully navigates the intersection without sharp deceleration, exploiting the limited space.

In this section, we compare the proposed method with widely used rule-based methods. We assigned 300 vehicles to approach the intersection at varying densities and measured the interval from when the first vehicle entered to when the last vehicle exited. Although our method entails a

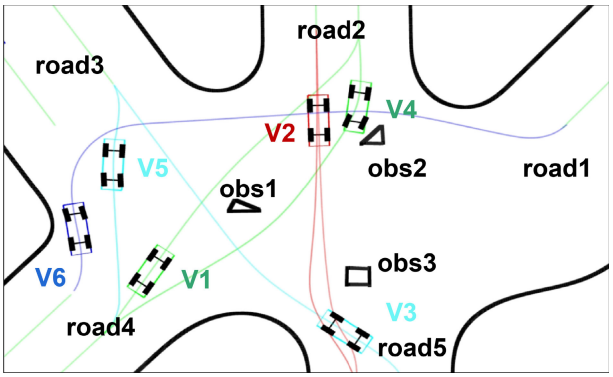


Fig. 15. Case 2: Different homotopy classes selection.

TABLE II
COMPARISON UNDER DIFFERENT TRAFFIC DENSITY

Traffic density	Test indicators	Rule-based	Our method
low density	Number of vehicle stops	137	0
	Maximum waiting time	17.6s	0s
	Average speed	7.13m/s	7.37m/s
	Total travel time	864.6s	767.5s
	Average CPU time	46ms	123ms
medium density	Number of vehicle stops	223	19
	Maximum waiting time	51.2s	4.8s
	Average speed	4.89m/s	6.95m/s
	Total travel time	891.4	467.6s
	Average CPU time	49ms	245ms
high density	Number of vehicle stops	267	53
	Maximum waiting time	92.3s	13.6s
	Average speed	3.37m/s	6.17m/s
	Total travel time	1130.5s	573.4s
	Average CPU time	53ms	312ms
uneven traffic flow	Number of vehicle stops	237	13
	Maximum waiting time	137.6s	15.7s
	Average speed	3.87m/s	6.26m/s
	Total travel time	967.6s	513.6s
	Average CPU time	52ms	356ms

higher computational burden, leading to longer computation times. However, the average computation time remains within 400 milliseconds. Assuming a driving speed of 10 meters per second, the vehicle would cover a distance of 4 meters within this time frame. Considering the buffer zone is significantly larger than this distance, the real-time performance meets operational requirements. Table II illustrates that the average single vehicle travel time for our proposed method is approximately 1.7 seconds, whereas the traditional rule-based method averages around 3.0 seconds. Low density traffic has a longer travel time than medium density traffic in our method, since the moment when the 300th vehicle enters the intersection is 750 seconds. The advantages of our method become more evident with increasing vehicle density.

In scenarios of uneven traffic flow, the maximum waiting time of rule-based methods tends to increase significantly, whereas our method remains largely unaffected. This discrepancy stems from high-density vehicle clusters converging at intersections during specific intervals, causing increased trajectory conflicts and frequent stops in rule-based methods. Conversely, our method effectively mitigates the challenge posed by concentrated vehicle entry at intersections. Given ample spatio-temporal resources at intersections, our method sustains high traffic efficiency with minimal impact.

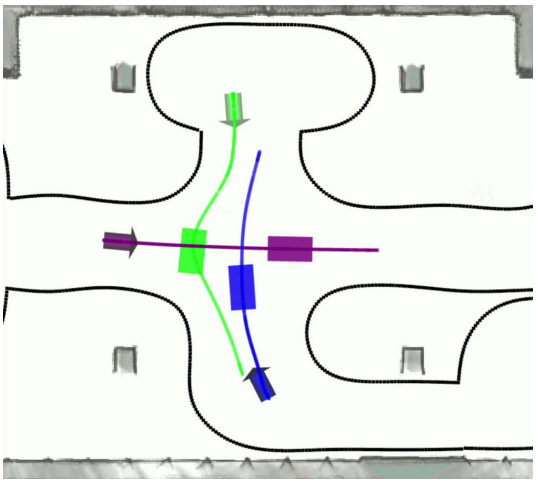


Fig. 16. Experiment map and vehicle trajectories.

Fig. 17. Real situation of the vehicles.

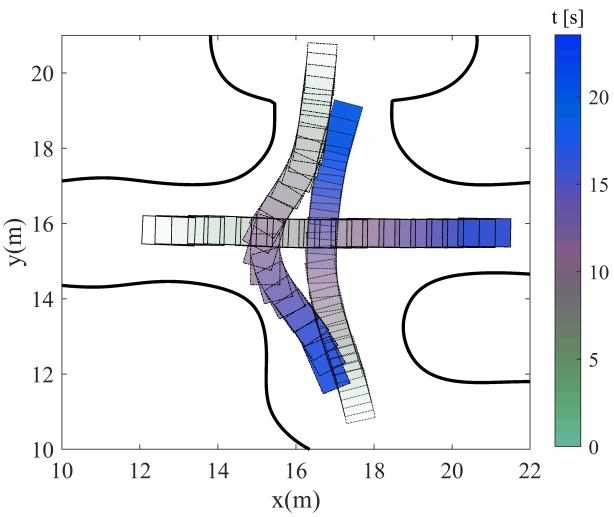


Fig. 18. Vehicle trajectories in experiment.

B. Experiment

To further validate the feasibility of the algorithm, we conducted real vehicle experiments using three 1.35m × 0.75m miniature mining trucks. For each vehicle, the velocity ranged from 0 to 1.0m/s, with a control frequency of 50Hz. The onboard 16-beam LiDAR was utilized for environment perception and vehicle localization. Communication between each vehicle and the scheduling node was facilitated through the Transmission Control Protocol (TCP) within the ROS system. The algorithm proposed in this article was implemented using C++ and executed on an industrial personal computer (IPC) equipped with an Intel i5-9400 CPU.

As depicted in Fig. 16, the experiments were conducted in a 13m × 13m simulated unstructured road scene, with a lane width of 2.5m for each road. The different colored lines on the map represent the driving trajectories of vehicles on distinct roads. The purple vehicle, with the highest priority, traversed the intersection at a constant speed. The blue vehicle, possessing lower priority, slightly adjusted its path shape to avoid collision with the purple vehicle. The green vehicle, with the lowest priority, navigated through the narrow space on the left side of the blue vehicle.

Fig. 17 displays the actual positioning of vehicles as introduced in Fig. 16. Observations show that, even during intense conflicts, vehicles maintained a sufficient safe distance. Fig. 18 depicts the actual trajectories of three vehicles at the intersection. Each vehicle is represented by a unique color shade, indicating its temporal order, with darker shades denoting closer proximity to the end point. Non-overlapping rectangles of the same color demonstrate the feasibility of generating safe, collision-free multi-vehicle trajectories, even accounting for control errors.

VI. Conclusion

The article introduces a framework combining priority-based centralized decision-making and V-Hybrid A* for distributed planning in conflict areas on unstructured roads. Vehicle priority is determined by entry time, decoupling conflicts between vehicles. Initially, each vehicle uses the V-Hybrid A* algorithm to identify unoccupied resource blocks at the intersection, generating an initial trajectory. Subsequently, box constraints are applied to each trajectory point, considering dynamic obstacles' collision avoidance, to optimize the path and ensure smooth trajectories. Finally, based on the initial trajectory, a convex space is created for speed planning, resulting in a final collision-free trajectory. Long-term traffic flow simulations show the trajectories and vehicle speeds generated meet traffic requirements. The search algorithms accurately provide homotopy classes and identify narrow spaces for collision avoidance. The proposed method significantly outperforms mature rule-based methods, reducing travel time and enhancing efficiency while satisfying real-time requirements. Real vehicle experiments confirm the algorithm's practical applicability.

Our future work includes: 1) Designing more comprehensive prioritization rules to better leverage the collaborative potential between vehicles. 2) Providing detailed solutions in case of planning failure and emergency parking to ensure efficient traffic at intersections.

References

[1] X. Chen, B. Xu, X. Qin, Y. Bian, M. Hu, and N. Sun, "Non-signalized intersection network management with connected and automated vehicles," *IEEE Access*, vol. 8, pp. 122065–122077, 2020.

[2] A. Benloucif, A.-T. Nguyen, C. Sentouh, and J.-C. Popieul, "Cooperative trajectory planning for haptic shared control between driver and automation in highway driving," *IEEE Trans. Ind. Electron.*, vol. 66, no. 12, pp. 9846–9857, Dec. 2019.

[3] M. R. Oudainia, C. Sentouh, A.-T. Nguyen, and J.-C. Popieul, "Personalized decision making and lateral path planning for intelligent vehicles in lane change scenarios," in *Proc. IEEE 25th Int. Conf. Intell. Transp. Syst. (ITSC)*, Oct. 2022, pp. 4302–4307.

[4] Y. Wang, X. Cao, and Y. Hu, "A trajectory planning method of automatic lane change based on dynamic safety domain," *Automot. Innov.*, vol. 6, no. 3, pp. 466–480, Aug. 2023.

[5] B. Xu et al., "Cooperative method of traffic signal optimization and speed control of connected vehicles at isolated intersections," *IEEE Trans. Intell. Transp. Syst.*, vol. 20, no. 4, pp. 1390–1403, Apr. 2019.

[6] B. Xu, X. J. Ban, Y. Bian, J. Wang, and K. Li, "V2I based cooperation between traffic signal and approaching automated vehicles," in *Proc. IEEE Intell. Vehicles Symp. (IV)*, Jun. 2017, pp. 1658–1664.

[7] K. Dresner and P. Stone, "Multiagent traffic management: A reservation-based intersection control mechanism," in *Proc. Joint Conf. Auton. Agents Multiagent Syst.*, vol. 3, 2004, pp. 530–537.

[8] A. I. M. Medina, N. van de Wouw, and H. Nijmeijer, "Cooperative intersection control based on virtual platooning," *IEEE Trans. Intell. Transp. Syst.*, vol. 19, no. 6, pp. 1727–1740, Jun. 2018.

[9] B. Xu et al., "Distributed conflict-free cooperation for multiple connected vehicles at unsignalized intersections," *Transp. Res. C, Emerg. Technol.*, vol. 93, pp. 322–334, Aug. 2018.

[10] Y. Zhang, R. Hao, T. Zhang, X. Chang, Z. Xie, and Q. Zhang, "A trajectory optimization-based intersection coordination framework for cooperative autonomous vehicles," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 9, pp. 14674–14688, Sep. 2022.

[11] Z. Hu, J. Huang, Z. Yang, and Z. Zhong, "Embedding robust constraint-following control in cooperative on-ramp merging," *IEEE Trans. Veh. Technol.*, vol. 70, no. 1, pp. 133–145, Jan. 2021.

[12] R. Hult, G. R. Campos, E. Steinmetz, L. Hammarstrand, P. Falcone, and H. Wymeersch, "Coordination of cooperative autonomous vehicles: Toward safer and more efficient road transportation," *IEEE Signal Process. Mag.*, vol. 33, no. 6, pp. 74–84, Nov. 2016.

[13] R. Hult, G. R. Campos, P. Falcone, and H. Wymeersch, "An approximate solution to the optimal coordination problem for autonomous vehicles at intersections," in *Proc. Amer. Control Conf. (ACC)*, Jul. 2015, pp. 763–768.

[14] T. Schouwenaars, B. De Moor, E. Feron, and J. How, "Mixed integer programming for multi-vehicle path planning," in *Proc. Eur. Control Conf. (ECC)*, Sep. 2001, pp. 2603–2608.

[15] D. Mellinger, A. Kushleyev, and V. Kumar, "Mixed-integer quadratic program trajectory generation for heterogeneous quadrotor teams," in *Proc. IEEE Int. Conf. Robot. Autom.*, May 2012, pp. 477–483.

[16] B. Li, H. Liu, D. Xiao, G. Yu, and Y. Zhang, "Centralized and optimal motion planning for large-scale AGV systems: A generic approach," *Adv. Eng. Softw.*, vol. 106, pp. 33–46, Apr. 2017.

[17] B. Li, Y. Ouyang, Y. Zhang, T. Acarman, Q. Kong, and Z. Shao, "Optimal cooperative maneuver planning for multiple nonholonomic robots in a tiny environment via adaptive-scaling constrained optimization," *IEEE Robot. Autom. Lett.*, vol. 6, no. 2, pp. 1511–1518, Apr. 2021.

[18] S. Glaser, B. Vanholme, S. Mammar, D. Gruyer, and L. Nouveliere, "Maneuver-based trajectory planning for highly autonomous vehicles on real road with traffic and driver interaction," *IEEE Trans. Intell. Transp. Syst.*, vol. 11, no. 3, pp. 589–606, Sep. 2010.

[19] I. Papadimitriou and M. Tomizuka, "Fast lane changing computations using polynomials," in *Proc. Amer. Control Conf.*, 2003, pp. 48–53.

[20] M. Werling, S. Kammel, J. Ziegler, and L. Gröll, "Optimal trajectories for time-critical street scenarios using discretized terminal manifolds," *Int. J. Robot. Res.*, vol. 31, no. 3, pp. 346–359, Mar. 2012, doi: 10.1177/0278364911423042.

[21] X. Li, K. Liao, and W. Chen, "Two-stage multi-AGV path planning based on speed pre-allocation," in *Proc. IEEE 3rd Adv. Inf. Manage., Communicates, Electron. Autom. Control Conf. (IMCEC)*, Oct. 2019, pp. 657–663.

[22] K.-D. Kim and P. R. Kumar, "An MPC-based approach to provable system-wide safety and liveness of autonomous ground traffic," *IEEE Trans. Autom. Control*, vol. 59, no. 12, pp. 3341–3356, Dec. 2014.

[23] C. E. Luis, M. Vukosavljev, and A. P. Schoellig, "Online trajectory generation with distributed model predictive control for multi-robot motion planning," *IEEE Robot. Autom. Lett.*, vol. 5, no. 2, pp. 604–611, Apr. 2020.

[24] C. K. Verginis and D. V. Dimarogonas, "Adaptive robot navigation with collision avoidance subject to 2nd-order uncertain dynamics," *Automatica*, vol. 123, Jan. 2021, Art. no. 109303.

[25] D. V. Dimarogonas, S. G. Loizou, and K. J. Kyriakopoulos, "A feedback stabilization and collision avoidance scheme for multiple independent non-point agents," *Automatica*, vol. 42, no. 2, pp. 229–243, 2006.

[26] G. Li, X. Zhang, H. Guo, B. Lenzo, and N. Guo, "Real-time optimal trajectory planning for autonomous driving with collision avoidance using convex optimization," *Automot. Innov.*, vol. 6, no. 3, pp. 481–491, Aug. 2023.

[27] K. Wang, Y. Wang, L. Wang, H. Du, and K. Nam, "Distributed intersection conflict resolution for multiple vehicles considering longitudinal-lateral dynamics," *IEEE Trans. Veh. Technol.*, vol. 70, no. 5, pp. 4166–4177, May 2021.

[28] M. Chen, J. F. Fisac, S. Sastry, and C. J. Tomlin, "Safe sequential path planning of multi-vehicle systems via double-obstacle Hamilton–Jacobi–Isaacs variational inequality," in *Proc. Eur. Control Conf. (ECC)*, Jul. 2015, pp. 3304–3309.

[29] D. R. Robinson, R. T. Mar, K. Estabridis, and G. Hewer, "An efficient algorithm for optimal trajectory generation for heterogeneous multi-agent systems in non-convex environments," *IEEE Robot. Autom. Lett.*, vol. 3, no. 2, pp. 1215–1222, Apr. 2018.

[30] W. Wu, S. Bhattacharya, and A. Prorok, "Multi-robot path deconfliction through prioritization by path prospects," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2020, pp. 9809–9815.

[31] H. Xu, S. Feng, Y. Zhang, and L. Li, "A grouping-based cooperative driving strategy for CAVs merging problems," *IEEE Trans. Veh. Technol.*, vol. 68, no. 6, pp. 6125–6136, Jun. 2019.

[32] A. Nakamura, Y.-C. Liu, and B. Kim, "Short-term multi-vehicle trajectory planning for collision avoidance," *IEEE Trans. Veh. Technol.*, vol. 69, no. 9, pp. 9253–9264, Sep. 2020.

[33] Y. Chen, M. Cutler, and J. P. How, "Decoupled multiagent path planning via incremental sequential convex programming," in *Proc. IEEE Int. Conf. Robot. Autom.*, Seattle, WA, USA, May 2015, pp. 5954–5961.

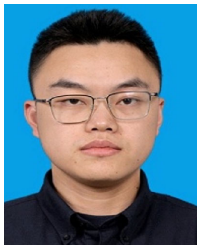
[34] P. Polack, F. Altché, B. d'Andréa-Novel, and A. de La Fortelle, "The kinematic bicycle model: A consistent model for planning feasible trajectories for autonomous vehicles?" in *Proc. IEEE Intell. Vehicles Symp. (IV)*, Jun. 2017, pp. 812–818.

[35] A. M. Shkel and V. Lumelsky, "Classification of the Dubins set," *Robot. Auto. Syst.*, vol. 34, no. 4, pp. 179–202, Mar. 2001.

[36] G. Chen et al., "Multiobjective scheduling strategy with genetic algorithm and time-enhanced A planning for autonomous parking robotics in high-density unmanned parking lots," *IEEE/ASME Trans. Mechatronics*, vol. 26, no. 3, pp. 1547–1557, Jun. 2021.

Biao Xu (Member, IEEE) received the B.E. and Ph.D. degrees from Tsinghua University, Beijing, China, in 2013 and 2018, respectively. From 2016 to 2017, he was a Visiting Scholar with the University of Washington, Seattle, WA, USA. He is currently an Associate Research Fellow with the College of Mechanical and Vehicle Engineering, Hunan University. His current research interests include connected and automated vehicles, vehicle control, and V2I cooperation. He was a recipient of the Best Paper Award in 14th Intelligent Transportation Systems Asia–Pacific Forum in 2015 and the Best Paper Award in the 2017 IEEE Intelligent Vehicle Symposium.

Guan Wang received the B.S. degree in automotive service engineering from Wuhan University of Science and Technology, Wuhan, China, in 2019. He is currently pursuing the M.S. degree in vehicle engineering with Hunan University, Changsha, China. His current research interests include vehicle motion planning, etc.



Zeyu Yang received the Ph.D. degree from the School of Vehicle and Mobility, Tsinghua University, Beijing, China, in 2021. He is currently an Associate Research Fellow with the College of Mechanical and Vehicle Engineering, Hunan University. His research interests include robust control, vehicle dynamics control, multi-vehicle cooperative control, and vehicle motion planning.

Yougang Bian (Member, IEEE) received the B.E. and Ph.D. degrees from Tsinghua University, Beijing, China, in 2014 and 2019, respectively. He was a Visiting Scholar with the Department of Electrical and Computer Engineering, University of California at Riverside, from 2017 to 2018. His research interests include distributed control, cooperative control, and their applications to connected and automated vehicles. He was a recipient of the Best Paper Award at the 2017 IEEE Intelligent Vehicles Symposium.

Xiaowei Wang received the M.E. degree in computer architecture from Northeastern University, Shenyang, China, in 2011, and the Ph.D. degree in computer science from the University of Göttingen in 2017. He is currently a Research Fellow with the College of Mechanical and Vehicle Engineering, Hunan University, Changsha, China. His research interests include vehicle-edge-cloud collaboration and intelligent dispatching.

Manjiang Hu received the B.Tech. and Ph.D. degrees from Jiangsu University, China, in 2009 and 2014, respectively. He was a Post-Doctoral Researcher with the Department of Automotive Engineering, Tsinghua University, from 2014 to 2017. He is currently a Professor with the College of Mechanical and Vehicle Engineering, Hunan University, Changsha, China. His research interests include cooperative driving assistance technology and vehicle control.