# Enhanced Techniques in Path Planning and Optimization for Autonomous Vehicles across Diverse Traffic Conditions

黄, 建宇

Kyushu University

Graduate School of Information Science and Electrical Engineering
Department of Advanced Information Technology

# Enhanced Techniques in Path Planning and Optimization for Autonomous Vehicles across Diverse Traffic Conditions

*By*

Huang Jianyu

*A thesis submitted in partial fulfillment of the requirements for the degree of Doctor of Engineering*

*Supervised by*

Professor Yutaka Arakawa

March 2024

**Huang Jianyu**

*Enhanced Techniques in Path Planning and Optimization for Autonomous Vehicles across Diverse Traffic Conditions*

**Supervisor:** Professor Yutaka Arakawa

**Advisory Committee:** Professor Yutaka Arakawa, Associate Professor Tsunenori Mine, and Associate Professor Shogo Fukushima

**Kyushu University**

Graduate School of Information Science and Electrical Engineering

Department of Advanced Information Technology

# Abstract

As technology advances, autonomous driving emerges as a crucial innovation in transportation, offering substantial prospects for Intelligent Transport Systems (ITS) and enhanced road safety. Despite these advancements, the evolution of autonomous driving technology presents new challenges in coping with intricate and dynamic traffic scenarios. Trajectory planning, a pivotal function in autonomous driving systems, dynamically orchestrates a continuous, smooth, and safe vehicle trajectory, considering vehicle dynamics, road geometry, and environmental changes in real time. This capability enables autonomous vehicles to navigate complex and dynamic road conditions while adhering to traffic regulations. However, when confronted with diverse and intricate traffic scenarios, a singular trajectory planning algorithm often struggles to achieve optimal performance across varied situations. To address this challenge, we focus on the diversity of autonomous driving traffic scenarios, particularly highlighting (1) adaptive cruise control with obstacle avoidance, (2) dense obstacle avoidance, and (3) lane-changing with obstacle avoidance scenarios in urban conditions. The paper provides a detailed analysis of the challenges posed by these scenarios to current mainstream planning algorithms and proposes corresponding modification methods.

(1) For the obstacle avoidance scenario of adaptive cruise control in urban environments, existing methods often overlook the effect of obstacle size on the safety of candidate trajectories during obstacle avoidance. This oversight results in undifferentiated obstacle avoidance actions when faced with large or small obstacles, leading to potential safety risks. In addition, the cost function for selecting the optimal trajectory does not consider the requirement to resist deviation from the road center line, often resulting in the vehicle deviating from the road center line after obstacle avoidance. To solve these problems, a multi-objective optimization algorithm is proposed using the Frenet framework to separate the lateral and longitudinal motion

of the vehicle. The algorithm comprehensively considers the comfort, safety, and deviation distance of the road center line. A novel quantification method that considers the size of obstacles is introduced to evaluate the safety of candidate trajectories. Experimental results show a significant improvement in comfort of 13.47%, 32.19%, 59.36%, and 18.60% in straight, curved, intersection, and U-shaped road scenarios, respectively. In addition, the algorithm improves anti-deviation by 63.72%, 13.86%, 44.36%, and 45.56% in the corresponding scenarios.

(2) In the context of dense obstacle avoidance scenarios, we propose a novel coarse-to-optimal path planning framework to address the low sampling effectiveness of conventional methods. Our approach introduces an adaptive sampling technique to construct a three-dimensional (3D) space-aware profile map within the Frenet framework, effectively quantifying abstract traffic scenarios. In the Frenet frame, the direction of vehicle forward motion is referred to as the $l$-direction, while the direction perpendicular to the $l$-direction is called the $s$-direction. This cost map guides waypoint generation by incorporating obstacle distributions in the $s$-direction and cost values in the $l$-direction. Subsequently, a designed cost function facilitates the acquisition of the coarse path, considering factors such as ride comfort, anti-deviation from the road center line, and path safety. The path model is then reconstructed based on the Taylor series, with quadratic programming fine-tuning the coarse path toward optimal trajectory generation. Experimental results demonstrate that our adaptive sampling method dynamically adjusts the sampling strategy based on changing obstacle numbers and distributions, significantly improving sampling efficiency to 100%, 99.85%, 99.68%, and 99.61% for scenarios with one, two, three, and four obstacles, respectively. In comparison to conventional methods, our approach achieves a 3.48% and 8.42% reduction in path length and an 18.68% and 30.21% reduction in maximum heading angle for scenarios with obstacles of the same and different sizes.

(3) For the lane-changing with obstacle avoidance scenario, we adopt a path-velocity decoupled (PVD) trajectory planning framework. In the path planning, to ensure the continuity of the lane-changing process and prevent the vehicle from deviating from the road center after lane change, we compare the performance of different curve models in lane-changing scenarios

and select the B-spline[1] curve as the path curve model. In terms of speed planning, to enhance real-time performance, we introduce a new method that directly generates a convex space, considering vehicle dynamics and traffic conditions. Experimental results show that, compared to the conventional method, our algorithm reduces the run time by 32.7%.

[1] https://en.wikipedia.org/wiki/B-spline

# Acknowledgements

In the process of completing this thesis, I would like to express my sincere gratitude to numerous individuals and organizations whose support has been instrumental in the successful completion of my research.

Firstly, I would like to express my sincere gratitude to the China Scholarship Council (CSC) for their generous support. Through the provision of a scholarship, they have given me the opportunity to pursue advanced studies in Japan, not only providing financial support but also instilling confidence in my academic potential. This encouragement has motivated me to strive for excellence.

Special thanks to Professor Arakawa for your meticulous guidance and unwavering support, which has been invaluable to my academic growth. Under your mentorship, I have acquired profound academic knowledge and cultivated a deeper understanding of research methodology. Your patience and trust have enabled me to continuously progress along the academic path. During the challenging period of COVID-19, your regular fortnightly meetings with Professor Ishida became a crucial source of support, alleviating my anxiety when I can't enter Japan. I appreciate your understanding and companionship during these difficult moments.

I am also very grateful to my parents. They have been the unwavering pillars of support throughout my academic journey, offering unconditional encouragement and understanding. Their diligent efforts and selfless dedication have made me feel incredibly fortunate.

I would also like to express my gratitude to my fellow students. Your camaraderie and support have added warmth and unity to the entire educational process. Every collaboration and discussion has greatly enhanced my research work and I sincerely appreciate your help.

I also owe a special debt of gratitude to myself. My perseverance in the face of difficulties, my unwavering commitment in the face of failures, and my constant efforts to improve have played a crucial role in shaping who I am today.

Finally, I would like to express my deepest gratitude to everyone I have met during my academic and life journey. Thank you for your support, which has enabled me to go further and pursue my dreams with unwavering determination.

Once again, I express my deepest appreciation to you all!

# Contents

# List of Figures

# List of Tables

# Acronyms

| | |
|---|---|
| ITS | intelligent transport systems |
| PVD | path-velocity decomposition |
| DP | dynamic programming |
| QP | quadratic program |
| CSC | China Scholarship Council |
| WHO | world health organisation |
| SAE | society of automotive engineers |
| HD | high-definition |
| GPS | global positioning system |
| IMU | inertial measurement unit |
| STJ | saptio-temporal joint |
| APF | artificial potential field |
| PRM | probabilistic road map |
| RRT | rapid-exploration random tree |
| CIL | conditional imitation learning |
| DRL | deep reinforcement learning |
| DQN | deep Q-network |
| DDPG | deep deterministic policy gradient |
| A3C | asynchronous advantage actor-critic |
| FPS | frames per second |

CIRL controllable imitation reinforcement learning

MAPPER multi-agent path planning with evolution and reinforcement learning

KSM kinematic single-track model

3D three-dimensional

# 1 Introduction

## 1.1 Background

With the continuous development of science and technology, automobiles have gradually become an irreplaceable means of transport for human traveling, making a great contribution to the communication and development of mankind, but at the same time bringing a lot of problems that cannot be ignored [1], [2]. With more and more vehicles traveling on urban roads, the ensuing congestion and traffic accidents are becoming more and more prominent, which not only affects people's traveling and the efficiency of social functioning, but also causes great loss of life and property, and therefore has gradually received more attention [3]. According to data released by the World Health Organisation (WHO), about 1.3 million people died in traffic accidents globally in 2022, which is equivalent to one death every 24.3 seconds[1].

With the development of industry, sensors capable of sensing information about the vehicle's surroundings and obstacles, as well as hardware devices such as central processors for processing such complex information, have become smaller and faster, and the overall automated control process of the vehicle is becoming more and more complete, which has made high-safety driverless technology possible [4]. A self-driving car is a vehicle that can dynamically and autonomously avoid a series of obstacles in the environment without relying on the conscious decisions of humans. According to the standards proposed by the Society of Automotive Engineers (SAE), autonomous driving can be divided into six levels: L0, L1, L2, L3, L4 and L5[2]. Among them, L0 means no automatic driving, L1 and L2 belong to the driver assistance stage, and L3, L4, and L5 belong to advanced automatic driving [5]. In

[1]https://www.who.int/zh/news-room/fact-sheets/detail/road-traffic-injuries
[2]https://www.sae.org/blog/sae-j3016-update

terms of the current technical reserves and industrial foundation, L1 and L2 have more application scenarios and markets, while L3 and above still have many technical difficulties [6], [7].

The autonomous driving system is functionally broadly comprised of a data source module, a perception module, a decision-making module, a path planning module, and a motion control module, as shown in Figure 1.1. The perception module seamlessly combines multiple data sources, including high-definition (HD) maps, cameras, radars, lasers, the global positioning system (GPS), and the inertial measurement unit (IMU) [8], [9], [10]. Its functions encompass tasks such as precise location determination, object detection, and tracking [11], [12]. The decision-making phase analyzes this data to predict trajectories and make decisions regarding driving strategies, which may include overtaking, lane changes, nudging, and vehicle following [13]. These decisions are made based on the information received from the perception module's data stream [14]. After establishing the chosen driving strategy, the planning module proceeds to craft a collision-free trajectory, taking into careful consideration the kinematic and dynamic constraints specific to the vehicle. This meticulous process ensures that the planned trajectory adheres to the vehicle's capabilities and characteristics, ultimately guaranteeing a safe and efficient path [15]. In the final stage, the system performs the complex task of calculating precise control signals for the steering wheel, accelerator, and brake pedals, enabling the vehicle to track the trajectory [16].

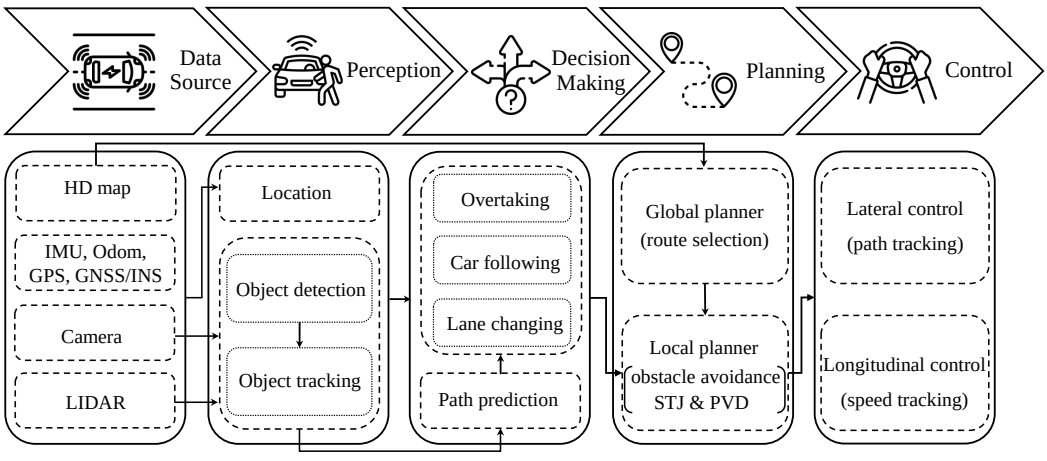



**Fig. 1.1:** An illustration of the autonomous driving system's architecture.

The planning module plays a key role within the autonomous driving system, not only acting as the critical link between the decision and control modules but also acting as a fail-safe mechanism in situations where the control module encounters difficulties in accurately following the intended path [17]. Its importance extends beyond mere coordination, as it acts as a safeguard to ensure the safe and reliable operation of the vehicle even when unexpected challenges arise during the journey. This means that the planning module provides redundancy and adaptability to deal with contingencies and maintain the overall integrity of the autonomous driving system, ensuring the safety and efficiency of the vehicle's autonomous navigation [18].

There are two planners in the planning module: global planner and local planner. The global planner is responsible for developing the vehicle's overall navigation route, taking into account many factors, including road conditions, traffic rules, geographic information, and navigation goals [19], [20]. Its output is a series of abstract path points that define the vehicle's approximate route from the start point to the destination. This global path provides a navigation guideline for the vehicle but does not focus on specific details such as the exact trajectory of the vehicle on the road [21].

Meanwhile, local path planning dynamically develops more specific, short-term paths based on global paths, based on real-time information about the vehicle's current location and surroundings, to ensure that the vehicle can cope with real-time changing situations, such as the sudden appearance of pedestrians, lane changes by other vehicles, road closures, etc. The output is a series of specific path points that define the actual trajectory of the vehicle and the tracking target of the control module [22].

These two planners complement each other. The global planner provides an overall navigation path for the vehicle, while the local planner fine-tunes the path to ensure that the vehicle can travel safely and efficiently. Together, the global and local planners form the trajectory planning module of an autonomous driving system, which is an important part of realizing autonomous driving. Through effective trajectory planning, the vehicle can avoid obstacles, comply with traffic rules, and adapt to changes in traffic conditions to achieve safe, smooth, and efficient autonomous driving.

This dissertation focuses on researching the problems of the local planner. The local planner includes three main planning frameworks: lateral-longitudinal decoupled trajectory planning, spatio-temporal joint (STJ) trajectory planning, and spatio-temporal decoupled trajectory planning, often referred to as PVD trajectory planning [23].

The lateral-longitudinal decoupled trajectory planning [24] aims to separate the processes of longitudinal and lateral motion in path planning, to increase system flexibility and performance. Longitudinal path planning focuses on the planning of forward speed and acceleration along the path, primarily addressing the need for appropriate speed adjustments under different road conditions and traffic scenarios to ensure safe and efficient travel. Lateral path planning considers the need for the vehicle to make lateral movements, such as turning, changing lanes, or avoiding obstacles, to ensure that the vehicle reaches its destination as expected. By decoupling the longitudinal and lateral aspects, the system becomes more adept at handling complex driving scenarios, such as situations requiring rapid speed changes or agile responses to obstacles. This decoupling design also facilitates the modularisation of the system, allowing the longitudinal and lateral planners to be designed and optimized relatively independently. Such a design philosophy is widely used in areas such as autonomous driving and robotic navigation to enhance system robustness and adaptability. The framework of the algorithm proposed in the Chapter 4 of the paper is based on this planning philosophy.

The STJ-based trajectory planning considers both time and space dimensions and computes feasible trajectories in a three-dimensional spatio-temporal space. It shows impressive performance in a variety of scenarios, closely resembling the decision-making of experienced human drivers. However, the intricate nature of STJ-based planning comes at the cost of considerable time and computational resources, limiting its applicability for real-time responses, especially in complex driving scenarios.

In contrast, PVD-based trajectory planning is akin to making decisions while driving a car: first deciding how to steer, whether to go straight, turn left, or turn right (path planning). Then you consider how to control the throttle and brakes, whether to accelerate, decelerate, or maintain a constant speed (speed planning). In this approach, the complete spatial path is first described. It then calculates the time and speed for each point on that path, resulting

in the final comprehensive trajectory. While this method may not always find the global optimum, two high-quality local optima are often sufficient for most scenarios. Furthermore, decoupling the decision and planning problems into two two-dimensional problems greatly improves tractability and optimisability [25]. It also encourages the division of labor, which is why many manufacturers have adopted this approach. The limitation of this framework is that during path planning, exact speed information is unavailable. Conversely, during speed planning, the path cannot be altered, potentially leading to poor coordination between path and speed [26]. The frameworks of the algorithms proposed in the fifth and sixth chapters are based on this planning philosophy.

## 1.2 Research Motivations

The essence of trajectory planning is an optimization problem, and its ultimate goal is to find an optimal trajectory that combines safety, compliance with traffic rules, comfort, efficiency, and followability. Firstly, a successful local planner needs to be adaptive and able to make intelligent trajectory planning according to the current environment and context. However, real-life traffic scenarios are varied. Each scenario may involve different road conditions, traffic flows, and other uncertainties requiring a variety of complexities to be addressed, making it difficult for a single algorithm to achieve excellent performance in different scenarios. Therefore, it is necessary to use different algorithms or combine multiple algorithms effectively to meet the needs of various driving scenarios. Secondly, the trajectory planning problem is a typical non-convex problem, and there is no globally optimal solution for non-convex problems, so how to transform a non-convex problem into a convex one by combining it with the current traffic scenarios is also a top research priority. In addition, there is interactivity, i.e., the behavior of other traffic participants is affected by the behavior of the self-vehicle. For example, in a lane-changing scenario, there is another vehicle approaching behind the target lane, whether the self-driving car rushes to merge into the lane before the rear vehicle arrives, or gives way to the rear vehicle and waits for the rear vehicle to pass before merging into the lane. Overly aggressive behavior will appear unsafe and affect the normal driving of other traffic participants;

while overly conservative behavior will appear unintelligent, resulting in a very inefficient passage of the self-driving car.

## 1.3 Contribution

Considering the complexity of the actual scenarios of autonomous vehicles, this dissertation proposes the corresponding trajectory planning algorithms for the cruise scenario, the static multi-obstacle scenario, and the lane-changing scenario, respectively.

In the case of the scenario of adaptive cruise control with obstacle avoidance, the contributions include the following points:

- We present an efficient algorithmic framework for trajectory planning. To represent the complex 3D motion of the vehicle, our method first employs the Frenet frame, which separates the motion of the vehicle into two orthogonal components: one for longitudinal motion along the driving guideline and the other for lateral motion perpendicular to the road center line. The state space is then discretized to generate initial and final position points for the path. Trajectory solution spaces for both longitudinal and lateral motion are generated by connecting the sampled endpoint conditions to the initial condition using quintic or quartic polynomials. These longitudinal and lateral trajectories are then combined to form sets of trajectories. From these, a subset of feasible trajectories is selected based on vehicle kinematics and collision avoidance constraints. Finally, the optimal trajectory is determined by minimizing a predefined cost function tailored to optimal path planning, taking into account factors such as comfort, safety, and road center line deviation. This approach is designed to facilitate effective trajectory planning.
- In assessing the safety of potential trajectories, we introduce the size of the obstacle as a unique parameter to be considered. We evaluate the safety of each candidate trajectory by measuring the safety loss through the variance of the Gauss-Laplace operator. This method provides an innovative approach to improve trajectory safety evaluation.

- We demonstrate the performance of our proposed algorithm in four traffic scenarios. The experimental results show that the algorithm can generate optimal paths for a straight road, curvy road, intersection, and U-shaped road, allowing autonomous vehicles to safely and comfortably avoid obstacles and complete the path planning from start to endpoint. On the straight road, the mean value of $J_{\text{jerk}}$ decreases by 13.47%, and the mean value of $J_{\text{offset}}$ decreases by 63.72%. On the curvy road, the mean value of $J_{\text{jerk}}$ decreases by 32.19%, and the mean value of $J_{\text{offset}}$ decreases by 13.86%. On the intersection scenario, the mean value of $J_{\text{jerk}}$ decreases by 59.36%, and the mean value of $J_{\text{offset}}$ decreases by 44.36%. On the U-shaped scenario, the mean value of $J_{\text{jerk}}$ decreases by 18.60%, and the mean value of $J_{\text{offset}}$ decreases by 45.56%.

In the case of the scenario of dense obstacle avoidance, our main contributions can be summarised as follows:

- We present a pragmatic path planning framework that begins by discretizing the travel space using a customized adaptive sampling technique to acquire waypoints. Then, we generate a series of paths by linking the sampled waypoints using piecewise quintic polynomials. An initial coarse solution for the optimal path is determined based on a predefined cost function. To overcome potential smoothness problems at the junctions between the piecewise curves, and to ensure that the final path isn't constrained by the quintic polynomial description, we incorporate a Taylor expansion to model the path. Finally, we use a quadratic program (QP)-based smoothing approach to refine and optimize the coarse solution, resulting in the generation of the optimal path. This comprehensive methodology improves path planning, particularly in addressing the challenge of smooth path transitions.

- Our approach introduces an adaptive algorithm designed to increase the sampling flexibility and to dynamically adjust the sampling parameters in response to changing traffic scenarios. This algorithm involves a transformation from the Cartesian frame to the Frenet frame, which efficiently characterizes vehicle motion on curved roads. Within the Frenet frame, we perform an obstacle influence analysis to gain insight into how obstacles affect the navigation of the autonomous vehicle. Based on this analysis, we discretize and sample the driving space in

the $l$-direction, while considering obstacle positions for discretization and sampling in the $s$-direction. A custom gate matrix is used to effectively control the output of the sampling points. This adaptive algorithm improves the adaptability of the sampling process, especially in scenarios with complex road geometries and different obstacles.

- We validate the effectiveness of our proposed path planning algorithm through a comparative analysis with existing methods. The experimental results highlight that, unlike conventional techniques, our adaptive path planning algorithm can dynamically adjust its sampling parameters based on the distribution of road obstacles. This dynamic adaptation significantly improves the efficiency of candidate path generation and results in a more effective solution space for optimal path selection. Furthermore, the generated optimal path is characterized by its shorter length and smoother trajectory compared to those generated by conventional methods. To sum up, these results highlight the potential of our approach in addressing the challenges of path planning for autonomous vehicles, demonstrating its superiority in terms of both efficiency and path quality.

In the case of the lane-changing scenario, we adopt the PVD-based framework to decouple the path planning and speed planning tasks. The contributions are as follows:

- We conducted a rigorous performance evaluation of various curve models in the context of lane-changing scenarios. After careful analysis, the B-spline curve emerged as the most appropriate choice for the lane-change path model. This choice was based on the need for continuous curvature and the essential criterion of zero curvature at both the starting and merging points of the path.
- For the speed planning phase, we anticipate the upcoming motion states of the ego vehicle by using its current motion state and kinematic properties. This predictive data effectively delineates the convex space used for velocity planning within the $s - T$ graph. Compared with the conventional method of dynamic programming (DP) and QP, the average run time decreases 32.7%.

## 1.4 Thesis Outline

The rest of this dissertation is organized as Figure 1.2 shows. In Chapter 2 we present an overview of related works. In Chapter 3, the definition of the Frenet frame and the transformation relation between the Frenet frame and the Cartesian frame are introduced. In Chapter 4, a novel algorithm considering the sizes of obstacles for the cruise scenario is proposed. In Chapter 5, we propose a practical algorithm to tackle the problem of sampling inflexibility and increase the sampling effectiveness. In Chapter 6, we compare the performance difference between different curve models for lane-change scenarios and propose a faster speed planning method compared with the conventional method which utilizes DP and QP algorithm.

**Fig. 1.2:** The overall structure and contents of the dissertation.

# 2 Related Literature

Currently, mainstream trajectory planning algorithms can be specifically divided into five major methods: artificial potential field-based methods, graph search-based methods, random sampling-based methods, discrete optimization-based, and learning-based methods. For trajectory planning algorithms, it is crucial to ensure their completeness and optimality [27]. Completeness means that if a path solution exists between the start and end points, it can always be found. Optimality means that among all feasible path solutions, the optimal one can be determined, typically the shortest path.

It's also important to ensure that the trajectory solutions obtained are collision-free, sufficiently smooth, and easy for the vehicle controller to track.

## 2.1 Artificial Potential Field-based Methods

The Artificial Potential Field (APF) method, introduced by Khatib [28], is a path planning algorithm based on the principles of potential energy fields. The basic concept of this algorithm treats autonomous vehicles as charged particles moving in space, influenced by various fields in the environment. These fields include the attractive force of the target and the repulsive force of obstacles. The autonomous vehicle's path is determined within this potential energy field, where it moves continuously in the direction of attraction toward the target while being pushed away from obstacles by the repulsive field.

The advantages of the APF algorithm are its ability to handle complex environments with multiple obstacles, strong real-time performance, and robustness. In addition, its simplicity makes it relatively easy to implement and debug.

Orozco-Rosas et al. [29] integrated membrane computing with a genetic algorithm and the artificial potential field method to create a path that is both safe and navigable. Compared to the traditional artificial potential field method, this approach significantly reduces path planning time. Lu et al. [30] combined the potential field method with the sigmoid curve approach and redesigned both the attractive and repulsive fields. Their results showed that this method improves vehicle stability and ride comfort, surpassing the performance of the conventional potential field-based method. Rasekhipour et al. [31] introduced a path planning approach that combines vehicle dynamics and optimal control with the artificial potential field method. By using artificial potential fields to construct obstacle and road boundary fields, this method skilfully handles various obstacles and road structures, ensuring the feasibility and optimality of the planned path. Huang et al. [32] proposed the APFE-RN method, which uses the artificial potential field method to assign different potential field functions to different obstacles and road boundaries. This method involves a grid-based representation of the drivable area of the road and assigns impedance values to each edge based on the potential field functions. It uses a local current comparison technique to find a collision-free path.

Although this algorithm has advantages, it has certain drawbacks, such as a tendency to get stuck in local optimal solutions. In certain scenarios, such as environments with very narrow passages or highly curved paths, the algorithm can exhibit oscillations. In addition, the algorithm is sensitive to factors such as the size, shape, and distribution of obstacles in the environment, requiring tuning and parameter adjustments.

## 2.2 Graph Search-based Methods

The graph search-based methods typically partition weighted graphs into lattice maps and perform searches based on specific rules to find optimal solutions. In 2009, Pivtoraiko et al. [33] introduced the concept of state lattices, which consist of nodes representing states and motion primitives connecting these nodes. State lattices discretize continuous spaces by creating a search graph. By searching for a sequence of motion primitives that move

from the initial state to the target state, real-world path planning problems can be addressed. Likhachev et al. [34] successfully applied the concept of state lattices to autonomous parking on unstructured road surfaces. For unstructured roads, regular state lattices can be used because they are easy to construct and their motion primitives are consistent. However, on structured roads, the use of regular state lattices may not fully reflect the actual characteristics of the road. Kushleyev et al. [35] addressed the problem of vehicle path planning on structured roads by constructing state lattices that match the road characteristics and allow real-time computation of motion primitives. They also introduced the time dimension into state lattices to handle dynamic obstacles. Building on this, Mcnaughton et al. [36] incorporated speed and acceleration information into state lattices to facilitate the search for lane-changing paths. Once state lattices are constructed, graph search methods can be used to find optimal paths.

In the 1950s, the algorithm named Dijkstra was first introduced, based on graph theory, to solve the problem of finding the optimal path in a weighted directed graph. This algorithm has the characteristics of breadth-first search and can identify globally optimal paths, but tends to explore many irrelevant nodes, resulting in low computational efficiency [37]. Peter Hart, Nils Nilsson, and Bertram Raphael [38] of the Stanford Research Institute introduced A*, which is based on the Dijkstra algorithm. A* incorporates heuristic functions to reduce the number of nodes explored and improve the efficiency of the algorithm while ensuring the discovery of globally optimal paths. However, the paths generated by this search algorithm often contain indistinguishable turning points, which are not conducive to autonomous vehicle tracking. The hybrid A* algorithm adds kinematic constraints to the A* algorithm. Unlike traditional graph search algorithms that only consider connectivity between nodes, the hybrid A* algorithm ensures that the planned path partially satisfies the tracking requirements. In autonomous parking functions, the Hybrid A* algorithm is often combined with Reed-Shepp curves to plan a reversible path and speed up the convergence of the algorithm [39]. To improve real-time motion planning for autonomous vehicles, Stentz et al. [40] introduced the D* algorithm, which has the property of global incremental planning. As autonomous vehicles move, the starting position for searches changes. The D* algorithm reuses search results from the previous start position, reducing redundant searches and improving computational

efficiency. To address the limitations of single-pass incremental algorithms such as ARA* and LPA*, Likhachev [41] and Koenig [42] independently proposed improved algorithms based on their respective research. These algorithms are known as AD* and D* Lite algorithms, and experimental results have shown that they achieve similar search efficiency to D*.

## 2.3 Random Sampling-based Methods

Random sampling-based methods have probabilistic completeness, which means that if a feasible solution exists on the map, it will be found by these algorithms. The principle behind these algorithms is to randomly place points within a region according to certain sampling rules. These points are then arbitrated based on cost values, with the highest cost node selected as the candidate node for the next step. Finally, these nodes are connected by collision-free links [43].

However, due to the random nature of this kind of algorithm, the generated paths do not obey the kinematic constraints of the vehicle. Therefore, they are not directly tractable. Typically, a smooth back-end optimization process is used after these algorithms have found an initial solution. In addition, these algorithms have strong search capabilities, but their blind sampling nature leads to a significant waste of computational resources [44].

Two typical algorithms in this category are the Probabilistic Road Map (PRM) and the Rapid-exploration Random Tree (RRT) [45]. The main idea of the PRM algorithm is to first randomly sample the entire map, creating a map of sample points. It then searches for the optimal solution within this map based on certain search cost criteria [46]. The RRT algorithm, on the other hand, doesn't need to create a map of sample points beforehand. Instead, it dynamically generates a search tree during the sampling process and backtracks to find an optimal path once a sample point is close to the destination. As RRT is a random sampling method and may not effectively converge to the goal, Kuffner et al. [47] proposed the RRT-connect algorithm. This algorithm simultaneously generates random trees from both the start and end points, significantly improving the search speed compared to basic RRT.

For planning algorithms based on random sampling, the challenge is to balance the trade-off between optimality and sampling density. Dealing with narrow passage problems is another major challenge for these algorithms. Li et al. [48] introduced a self-adaptive RRT algorithm to address these issues. It used a greedy search strategy to speed up convergence and integrated an environment evaluation module during exploration to help the algorithm navigate through narrow passages. Feng [49] also presented an algorithm for autonomous driving motion planning based on RRT. This algorithm utilized a parameterized node generation method to address problems arising from RRT-generated paths that don't respect vehicle kinematic constraints. It introduced strategies such as endpoint tree policy and termination condition evaluation to improve real-time performance. Finally, considering the random sampling nature of RRT, it generated a guidance domain using A* and constrains RRT's sampling space to improve solution quality.

## 2.4 Discrete Optimization-based Methods

Discrete optimisation-based methods are currently widely used. Since the essence of trajectory planning is a multi-objective optimization problem, a general approach is to systematically sample structured roads, connect these sampled points using curve models, such as polynomial curves, Bezier curves, and B-spline curves, etc., define objective functions based on real scenarios or operating conditions, and then transform information from the autonomous vehicle's perception layer, such as lane markings, obstacle data, and vehicle system dynamic constraints, into equality or inequality constraints. This transformation transforms the trajectory planning problem into an optimization problem. The optimal trajectory is then determined by iterative computation.

Hu et al. [50] developed a Gaussian convolution-based cost function that takes into account the relationship between local paths and obstacle safety. This method allows the selection of multiple paths, taking into account both stationary and moving obstacles. However, it doesn't effectively integrate vehicle speed into path selection, doesn't provide the global optimum, and is highly dependent on sampling point density and parameter tuning, which

may limit its adaptability to complex scenarios. Werling et al. [51] integrated decision behavior with path planning. They decomposed the feasible range of the vehicle into lateral and longitudinal dimensions and fitted changing curves over time for both dimensions. A cost function prioritizes the trajectory ranking, followed by collision checking based on priority. Trajectories that fail the collision constraint are discarded, while those that meet the criteria are output. However, the adaptability of the algorithm in complex environments requires further investigation. Baidu Apollo's Fan et al. [52] developed a decision planning system aimed at solving industrial planning problems. This system considers safety, convenience, and scalability. It iteratively solves the path and speed optimization problem based on the Frenet framework, while simultaneously managing traffic rules, and obstacle decisions, and generating smooth trajectories. The complexity of this algorithm is high and it doesn't explicitly state when the solution is optimal. It also requires significant parameter tuning. Zhang et al. [53] proposed a path planning algorithm based on quadratic programming. They also introduced a piecewise jerk method to describe vehicle kinematic constraints in the Frenet coordinate system. By decoupling the lateral and longitudinal directions and transforming the nonlinear programming problem into a quadratic programming problem, a faster solution is achieved. Zhou et al. [54] introduced a self-driving trajectory planning algorithm suitable for open spaces. They decouple the trajectory planning problem into two planning problems: one for the lateral (horizontal) direction and another for the longitudinal (vertical) direction. They solve these problems independently using the DL-IAPS and PJSO algorithms. Wonteak Lim et al. [55] introduced a hybrid trajectory planning algorithm that exploits the advantages of both sampling-based and numerical optimization-based methods. It has shown promising results in dealing with structured road environments with dynamic conditions and complex obstacles.

## 2.5 Learning-based Methods

In recent years, with the advancement of artificial intelligence, in particular, the rise of deep neural network technologies, learning-based path planning methods have emerged as a promising approach to solving path planning

problems. Learning-based methods establish end-to-end models that bridge the gap between perception and control [56]. They unify various driving subtasks such as scene perception, object recognition, target tracking, and planning decisions into deep neural networks [57]. These networks map perceptual information directly to control signals such as throttle, steering, and braking, achieving an integrated cognitive-to-control decision process. This approach eliminates the need for module separation, simplifies the cumbersome task of feature engineering, and streamlines the structure of autonomous driving systems for improved efficiency [58]. Based on the principles of deep neural network learning, learning-based methods can generally be divided into two main types: imitation learning and deep reinforcement learning [59].

### 2.5.1 Imitation Learning Methods

In the field of autonomous driving, imitation learning has become a topic of significant interest among researchers. Imitation learning focuses on harnessing the powerful nonlinear approximation capabilities of deep neural networks to learn expert driving strategies [60]. Typically, imitation learning is performed through supervised learning. Before training, an expert data set is generated and used for training. The algorithm learns these strategies from the dataset and iterates to reach an optimal policy. The input data for training includes information collected from sensors such as cameras, GPS, and LiDAR. The training objective is to predict the control signals, such as throttle, steering, and braking, at the same time as an expert driver performs these actions [61].

Pomerleau [62] introduced the ALVINN network in 1989, marking the initial concept of end-to-end imitation learning. This network consisted of a three-layer backpropagation fully connected neural network with one hidden layer, and it was applied to road tracking tasks. Frossard et al. [63] proposed a three-dimensional trajectory detection and target tracking method using multiple sensors such as cameras and LiDAR to collect expert data for imitation learning. This algorithm primarily addresses the task of perceiving traffic participants during the process of autonomous driving. The algorithm takes filtered LiDAR and camera data as input and predicts the future states

of objects detected in the scan over a period of time. The core idea of this algorithm is to learn object detection and matching using convolutional neural networks, and its feasibility will ultimately be demonstrated through practical experiments. Ross et al. [64] introduced a data aggregation algorithm, called DAGGER, to address the problem of error accumulation in imitation learning due to the lack of representative original driving data. This algorithm attempts to collect expert demonstration data under the state distribution induced by the learning policy. It involves generating new examples by adding expert-corrected annotations to the current policy and incorporating them into the demonstration dataset for retraining. However, re-annotating data with only a single image is particularly challenging and lacks consideration of temporal continuity. He et al. [65] introduced a coaching strategy, which is used to optimize the relationship between the learning speed and the strategy space in DAGGER. This strategy constructs an easily learnable policy, allowing for gradual convergence to the optimal solution. Codevilla et al. [66] proposed Conditional Imitation Learning (CIL). In complex traffic situations, CIL introduces passenger commands into the network to make road choices. However, this approach suffers from the need for human intervention and doesn't effectively extract semantic information from the original image data. This results in low training efficiency and unstable driving.

In summary, although imitation learning methods offer significant advantages, such as the ability to learn quickly from expert strategies, their reliability is highly dependent on the quality of expert data. Although many research institutions and companies have provided access to complex and diverse datasets for research purposes, these datasets still cannot cover all scenarios. As a result, there are still challenging scenarios in research.

### 2.5.2 Deep Reinforcement Learning Methods

Reinforcement learning acquires robust driving strategies through continuous interaction between the agent and the environment [67]. This approach relies on the construction of appropriate reward functions to evaluate the agent's behavior and provide appropriate reward or penalty signals, allowing continuous adjustment and optimization of network parameters to maximize

cumulative rewards, ultimately achieving task-oriented control strategy learning [68]. However, for control tasks where there is no prior knowledge and no theoretical guidance for the design of reward functions, reinforcement learning methods often struggle to achieve the desired performance. Learning a basic strategy from scratch can be time-consuming due to the initial lack of task-specific knowledge. Deep Reinforcement Learning (DRL) is a subfield that combines deep learning methods with reinforcement learning, using deep neural networks to represent and approximate complex value functions and policies to solve problems with high-dimensional state and action spaces [69]. Typical representatives of DRL are Deep Q-Network (DQN) [70], and Deep Deterministic Policy Gradient (DDPG) algorithms [71].

Dosovitskiy et al. [72] conducted training on the CARLA simulator using the Asynchronous Advantage Actor-Critic (A3C) algorithm with 10 million simulated steps, equivalent to about 12 days of training at 10 frames per second (FPS). Although this algorithm showed good performance in straight-line tasks, its performance in other tasks, such as turning and intersection selection, was suboptimal. In contrast, Toromanoff et al. [73] introduced a technology that pre-extracts high-level road information from images, enabling them to train a vision-driven reinforcement learning agent with over 20 million iterations in the urban environment of CARLA. This agent can interact with traffic lights and manage road junctions. Liang et al. [74] also proposed Controllable Imitation Reinforcement Learning (CIRL). They first trained an agent through imitation learning within a reasonably constrained action space and then fine-tuned it using the Deep Deterministic Policy Gradient (DDPG) optimization algorithm. This approach successfully learned driving strategies in a high-fidelity simulator by integrating imitation learning with reinforcement learning. However, most of the performance improvement was attributed to the pre-training phase using imitation learning. Yang et al. [75] used the Deep Q-Network (DQN) algorithm in deep reinforcement learning to address multi-robot path planning problems. This algorithm combines Q-learning, experience replay mechanisms, and volume-based neural network techniques to generate target Q-values. Liu et al. [76] introduced a distributed, partially observable, multi-agent path planning approach using the Multi-Agent Path Planning with Evolution and Reinforcement Learning (MAPPER) method. It learns effective local planning strategies in mixed dy-

namic environments using image-based representations and training policy non-homogeneity assumptions.

## 2.6 Summary

In summary, trajectory planning algorithms for autonomous driving can be divided into two main types: algorithm-based methods and end-to-end models based on machine learning. Algorithm-based methods often take a layered and decoupled approach, integrating the results of different algorithms to achieve better performance. For example, in open-space trajectory planning algorithms, after using graph-based algorithms to find feasible solutions, optimization algorithms are used for secondary optimization. This ensures that the final output trajectory meets comfort requirements and allows for fine obstacle avoidance manoeuvres. In addition, geometric curve-based algorithms can be combined with graph search and sampling algorithms to improve convergence rates. In structured road environments, dynamic programming algorithms can be integrated with results from secondary optimization in $s-L$ and $s-T$ coordinate systems to serve as the output of the planning module.

End-to-end autonomous driving models, on the other hand, involve fewer process decouplings, making them better suited to handle complex traffic scenarios with increased flexibility and optimality. However, the limited interpretability of machine learning algorithms makes it difficult to accurately modify them for specific problems, which hinders iterative development. As a result, end-to-end models have seen limited practical application and are primarily in the experimental exploration stage.

# 3 Fundamental Concepts

## 3.1 Criteria of Trajectory Planning

Trajectory planning algorithms are a key component among the various algorithms used in autonomous vehicles. They directly output the vehicle's reference trajectory, providing a desired path for the vehicle's lower-level linear controllers. These linear controllers calculate the appropriate throttle, brake, and steering wheel angle based on the reference trajectory to guide the vehicle along this path. Deploying a trajectory planning algorithm for practical testing on autonomous vehicles typically involves meeting specific criteria.

- Safety: most autonomous vehicles today are designed to carry people, so the safety of passengers and other traffic participants is of utmost importance. Even if the vehicle is not designed to carry people, the safety of the property should not be neglected. Therefore, safety must be the first reference index of the path planning design criteria.
- Smoothness: the reference trajectory output from neighboring frames must be nearly continuous, so that the trajectory can be coherent when executed by the linear controller, making the vehicle run steadily. If the output reference trajectory jumps obviously, and the previous frames of the linear controller execute actions that are very different from the current frames, it will directly lead to the vehicle swaying from side to side, and although the trajectory of each planning is reasonable, the comfort is poor. Therefore, when designing the trajectory planning algorithm, it is necessary to constrain the output in the time dimension so that it changes continuously and progressively as much as possible.
- Feasibility: the reference trajectory generated by a trajectory planning algorithm must be convertible into control inputs for the execution

modules (brake, throttle, and steering wheel angle) via the linear controller. Reference trajectories that cannot be executed by the vehicle are essentially meaningless. For instance, if the generated reference trajectory has a high curvature, implying that the vehicle must make a sharp turn, it may not be physically feasible. Hence, feasibility is a significant concern that trajectory planning algorithms must address.

- Optimality: while there can be numerous potential paths for an autonomous vehicle, typically only the optimal one is selected based on specific criteria. The criteria can vary depending on different scenarios and may be adjusted accordingly. However, the evaluation standards often include factors like smoothness, safety, etc.
- Anticipatory: when designing and developing trajectory planning algorithms, it is essential to consider the motion characteristics of surrounding vehicles or other dynamic objects. This involves modeling and predicting the motion state of these vehicles, including lane changes, acceleration, and deceleration, to assess potential future states over the next few seconds. The trajectory of an autonomous vehicle is then designed and planned based on these predictions.

## 3.2 Vehicle Kinematics Model

The vehicle kinematic model primarily describes the evolution of a vehicle's position and velocity over time from a geometric perspective, without considering the physical properties of the vehicle or the external forces acting on it. In the context of trajectory planning for autonomous driving, proper consideration of the vehicle model serves the purpose of achieving smoother trajectories and better trajectory tracking [77]. For example, during the trajectory generation phase, the vehicle kinematic model is of paramount importance as it ensures that the generated trajectories obey the vehicle kinematic constraints, such as maximum speed, maximum acceleration, minimum turning radius, etc [78]. Furthermore, the vehicle kinematic model not only helps to ensure that the generated trajectories comply with the vehicle kinematic constraints, but also assists in obstacle avoidance. Trajectory

planning must take into account the dimensions of the vehicle and potential obstacles along the path to generate safe trajectories [79]. Throughout this process, the kinematic model of the vehicle incorporates the geometric characteristics and kinematic constraints of the vehicle, ensuring that the generated trajectories skilfully navigate around obstacles to ensure safety during the journey [80].

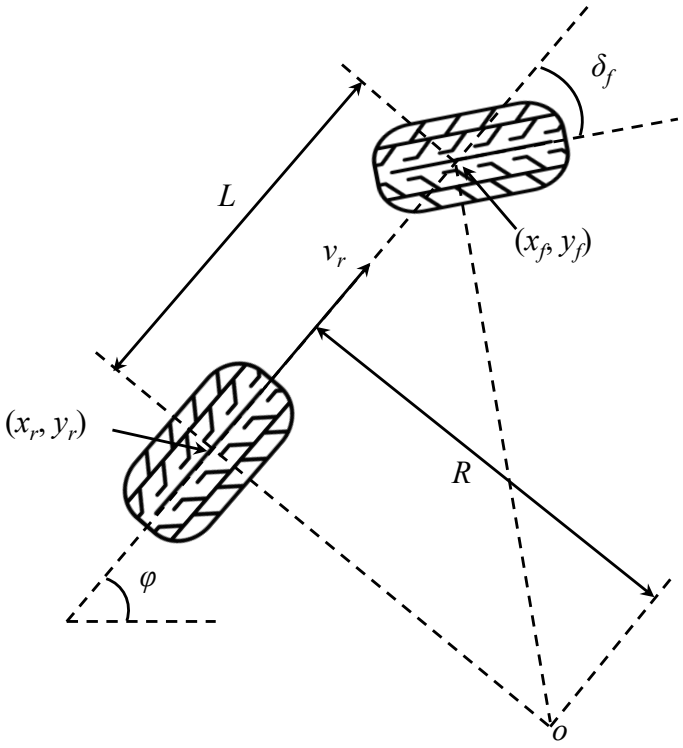



**Fig. 3.1:** Kinematic single-track model of the autonomous vehicle. $\varphi$ is the heading angle of the vehicle; $\delta_f$ is the front wheel angle, $(x_f, y_f)$, $(x_r, y_r)$ are the axial coordinates of the front and rear axes of the vehicle; $v_r$ is the longitudinal speed of the vehicle which is also the speed of the vehicle's rear axle; $L$ is the vehicle's wheelbase.

Among the many vehicle models, the most representative and used vehicle model is the Kinematic Single-Track Model (KSM) [81], as shown in Figure 3.1, which relies on a number of following assumptions:

- The vehicle follows a non-integrity constraint, i.e. the vehicle moves forward without the wheels slipping at the point of contact with the ground, but is free to rotate about its axis of rotation. The front wheels of the vehicle have an additional degree of freedom to move about an axis perpendicular to the plane of motion.

- The vehicle moves in a two-dimensional plane, ignoring the vertical motion of the vehicle.
- The vehicle suspension system is rigid and the suspension motion of the vehicle is not considered.
- The vehicle suspension system is rigid and does not take into account the suspension motion of the vehicle.
- During steering, the turning radius of the vehicle is the same as the radius of curvature of the road.
- The front wheels of an autonomous vehicle are responsible for steering and the rear wheels are responsible for driving.

Within a very short time interval $\Delta t$, it can be approximated that the vehicle moves in the direction of its body. Here, $d_y$ and $d_x$ represent the distances the vehicle travels along the $y$ and $x$ axes, respectively, over the time interval $d_t$ [82].

$$\frac{dy}{dx} = \frac{\dot{y}}{\dot{x}} = \tan\varphi \tag{3.1}$$

The front and rear axles of a vehicle have the following equation relationship:

$$\begin{cases} \dot{x}_f \sin\left(\varphi - \delta_f\right) - \dot{y}_f \cos\left(\varphi - \delta_f\right) = 0 \\ \dot{x}_r \sin(\varphi) - \dot{y}_r \cos(\varphi) = 0 \end{cases} \tag{3.2}$$

According to Figure 3.1, the velocity at the center of the rear axle $(x_r, y_r)$ can be presented as follows:

$$v_r = \dot{x}_r \cos(\varphi) + \dot{y}_r \sin(\varphi) \tag{3.3}$$

Combining Equations 3.2 and 3.3, it can be inferred as:

$$\begin{cases} \dot{x}_r = v_r \cos(\varphi) \\ \dot{y}_r = v_r \sin(\varphi) \end{cases} \tag{3.4}$$

Furthermore, the front and rear wheels of the vehicle have the following relationships:

$$\begin{cases} x_f = x_r + L\cos(\varphi) \\ y_f = x_r + L\sin(\varphi) \end{cases} \tag{3.5}$$

Combining equations (2.3), (2.4), and (2.5), it can be inferred as:

$$w = \frac{v_r}{L}\tan(\delta_f) \tag{3.6}$$

where $w$ is the yaw speed of the vehicle.

Based on Equation 3.6 and the vehicle speed, the vehicle steering radius $R$ and the front wheel angle $\delta_f$ are shown as:

$$\begin{cases} R = \frac{v_r}{w} \\ \delta_f = \tan^{-1}\frac{L}{R} \end{cases} \tag{3.7}$$

According to the Equations 3.4 (2. 4) and 3.6, the vehicle kinematics model is denoted as:

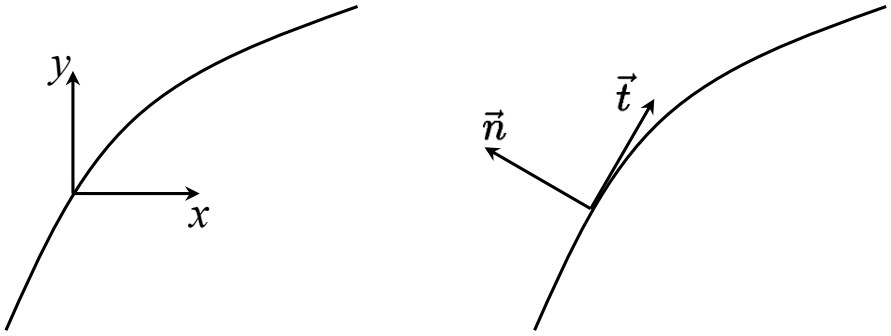
$$\begin{bmatrix} \dot{x}_r \\ \dot{y}_r \\ \dot{\varphi} \end{bmatrix} = \begin{bmatrix} v_r\cos(\varphi) \\ v_r\sin(\varphi) \\ v_r\tan(\delta_f)/L \end{bmatrix} \tag{3.8}$$

## 3.3 Frenet Frame

### 3.3.1 Introduction to the Frenet Frame

In the development and testing of autonomous driving algorithms, the most commonly used coordinate systems include the odometry frame, the Cartesian frame, and the Frenet frame [83], [84]. Describing the motion trajectory in the Frenet frame is only related to the choice of the reference line and is independent of the vehicle's absolute position [85]. Decomposing the motion trajectory of an autonomous vehicle into two directions related to the road geometry simplifies the trajectory planning model, making it easier to solve

the state differential equations and improving computational efficiency [86], [87]. The Frenet frame is shown as Figure 3.2




(a) Cartesian Frame (b) Frenet Frame

**Fig. 3.2:** An illustration of Cartesian frame and Frenet frame.

The Frenet-Serret formulas are used to describe the motion of a particle in Euclidean space on a continuous differentiable curve[1]:

$$s(t) = \int_0^t \|r'(\sigma)\| \, \mathrm{d}\sigma \tag{3.9}$$

where, $\vec{t}$ is the unit tangent vector at a point, i.e., the unit vector of the point along the direction of motion of the curve; $\vec{n}$ is the unit normal vector at a point, i.e., the unit vector of the point perpendicular to the direction of motion; $\vec{r}(t)$ is a non-degenerate curve in two-dimensional Euclidean space with time $t$; $s(t)$ is the cumulative length of the motion of a point on the curve at time $t$.

$s(t)$ is a strictly monotonically increasing function, and $t$ can be expressed as a function of $s$, therefore, the curve $\vec{r}$ can be expressed as a function of the cumulative length $s$:

$$\vec{r}(s) = \vec{r}(t(s)) \tag{3.10}$$

Then, the corresponding tangent vector $\vec{t}$, normal vector $\vec{n}$ can be expressed as:

$$\vec{t} = \frac{\frac{d\vec{r}}{ds}}{\left\|\frac{d\vec{r}}{ds}\right\|} \tag{3.11}$$

[1] https://en.wikipedia.org/wiki/Frenet-Serret_formulas

$$\vec{n} = \frac{\frac{d\vec{t}}{ds}}{\left\| \frac{d\vec{t}}{ds} \right\|} \tag{3.12}$$

Based on the above definition, the Frenet-Serret formula can be expressed as:

$$\frac{d\vec{t}}{ds} = K\vec{n} \tag{3.13}$$

$$\frac{d\vec{n}}{ds} = -K\vec{t} \tag{3.14}$$

$K$ is the curvature of the curve $r(s)$ and the Frenet-Serret formulas are finally defined in matrix form as shown in the following equation:

$$\begin{bmatrix} \vec{t}' \\ \vec{n}' \end{bmatrix} = \begin{bmatrix} 0 & K \\ -K & 0 \end{bmatrix} \begin{bmatrix} \vec{t} \\ \vec{n} \end{bmatrix} \tag{3.15}$$

### 3.3.2 Cartesian Frame to Frenet Frame

As shown in Figure 3.3, the position of the autonomous vehicle in the Cartesian frame and the Frenet frame is illustrated [88]. The establishment of the Frenet frame is based on a given reference line $T_{\text{ref}}$. $T_{\text{ref}}$ can be any curve, typically defined as the center line of the road lane [89].

Assuming that the coordinates of the autonomous vehicle in the Cartesian frame are $(x, y)$, a projection is made from the vehicle's position $(x, y)$ to the reference line $T_{\text{ref}}$, resulting in the point $P$. The distance between point $P$ and the vehicle's position represents the lateral displacement $l$. The curve distance from the starting point of the reference line to the projection point $P$ represents the longitudinal displacement $s$.

In the Cartesian Frame, vehicle motion is generally described as $[\vec{x}, \theta_x, \kappa_x, v_x, a_x]$, where, $\vec{x}$ corresponds to the position of the vehicle at point $Q(x, y)$; $\vec{n}_x$ and $\vec{t}_x$ represent the unit normal and tangent vectors of the vehicle's path at point $Q$; $\theta_x$ is the angle between $\vec{x}$ and the $x$-axis; $\kappa_x$ characterizes the curvature at point $Q$; $v_x$ indicates the velocity of the ego vehicle and $a_x$ stands for the vehicle's acceleration.

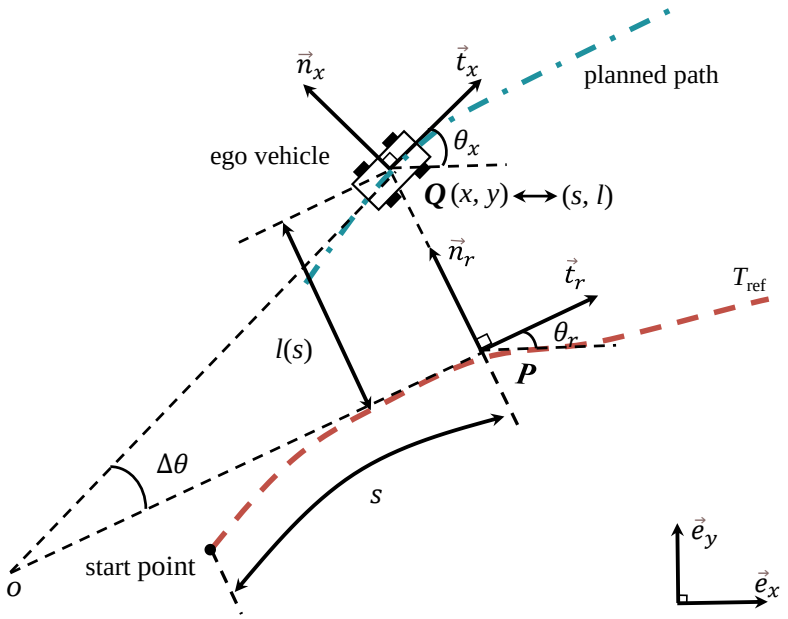



**Fig. 3.3:** Transformation between Frenet frame and Cartesian frame.

On the other hand, in a Frenet frame, the point $P$ represents the projection of the vehicle's position point $Q$ onto the reference line. The angle between $\vec{r}$ and the $x$-axis is denoted by $\theta_r$. The angle between $\vec{r}$ and the $\vec{x}$ is denoted by $\Delta\theta$ (i.e., $\Delta\theta = \theta_x - \theta_r$). $\vec{n}_r$ and $\vec{t}_r$ denote the unit normal and tangent vectors of the reference line at point $P$. The vehicle's status can typically be described using the vector $\left[s, \dot{s}, \ddot{s}, d, \dot{d}, \ddot{d}, d', d''\right]$, where the distance between $P$ and $Q$ corresponds to the transverse displacement $d$, and the curve distance from the start point of the reference line to $P$ represents the longitudinal displacement $s$. Additionally, $\dot{s}$ is the first derivative of $s$ with respect to time $t$ (i.e., $\dot{s} = \frac{ds}{dt}$), meanwhile $\ddot{s}$ is the second derivative of $s$ with respect to time $t$ (i.e., $\dot{s} = \frac{d\dot{s}}{dt}$). $d'$ denotes the first derivative of $d$ with respect to $s$ (i.e., $d' = \frac{dd}{ds}$), while $d''$ denotes the second derivative of $d$ with respect to $s$ (i.e., $d'' = \frac{dd'}{ds}$). $\dot{d}$ is the first derivative of $s$ with respect to time $t$ (i.e., $\dot{d} = \frac{dd}{dt}$), meanwhile $\ddot{d}$ is the second derivative of $d$ with respect to time $t$ (i.e., $\dot{d} = \frac{d\dot{d}}{dt}$).

According to the definition of the Frenet frame, the point $P$ is the nearest point to the point $Q$ on the reference line, then the $s$ coordinate of point $Q$ is:

$$s = s_r \tag{3.16}$$

It can be derived from Figure 3.3 and the definition of orthogonal bases:

$$\vec{t}_r = \begin{bmatrix} \cos\theta_r & \sin\theta_r \end{bmatrix}^T \tag{3.17}$$

$$\vec{n}_r = \begin{bmatrix} -\sin\theta_r & \cos\theta_r \end{bmatrix}^T \tag{3.18}$$

$$\vec{t}_x = \begin{bmatrix} \cos\theta_x & \sin\theta_x \end{bmatrix}^T \tag{3.19}$$

$$\vec{n}_x = \begin{bmatrix} -\sin\theta_x & \cos\theta_x \end{bmatrix}^T \tag{3.20}$$

Based on the vector relationship, we can get Equation 3.21:

$$\vec{x}(s,l) = \vec{r}(s) + l(s)\vec{n}_r(s) \tag{3.21}$$

In the following text, we use $\vec{r}$ to replace $\vec{r}(s)$, and use $l$ to replace $l(s)$.

By transforming Equation 3.21, we can obtain Equation 3.22:

$$\begin{aligned} l &= (\vec{x} - \vec{r})^T \vec{n}_r = \|\vec{x} - \vec{r}\|_2 \cos\left(\Delta\theta - \left(\theta_r + \frac{\pi}{2}\right)\right) \\ &= \|\vec{x} - \vec{r}\|_2 \left(\sin(\Delta\theta)\cos(\theta_r) - \cos(\Delta\theta)\sin(\theta_r)\right) \end{aligned} \tag{3.22}$$

where $\vec{x} = (x_x, y_x)$, $\vec{r} = (x_r, y_r)$, then $\|\vec{x} - \vec{r}\|_2$ can be represented as:

$$\|\vec{x} - \vec{r}\|_2 = \sqrt{(x_x - x_r)^2 + (y_x - y_r)^2} \tag{3.23}$$

Therefore, Equation 3.22 can be represented as follows:

$$l = \text{sign}\left((y_x - y_r)\cos(\theta_r) - (x_x - x_r)\sin(\theta_r)\right)\sqrt{(x_x - x_r)^2 + (y_x - y_r)^2} \tag{3.24}$$

According to the definitions of $v_x$ and $\dot{s}$:

$$\dot{\vec{r}} = \dot{s}\vec{t}_r \tag{3.25}$$

$$\dot{\vec{x}} = v_x \vec{T}_x \tag{3.26}$$

According to Equations 3.21, 3.25 and 3.26:

$$\begin{aligned} \dot{l} &= (\dot{\vec{x}} - \dot{\vec{r}})^T \vec{n}_r + (\vec{x} - \vec{r})^T \dot{\vec{n}}_r \\ &= v_x \vec{t_x}^T \vec{n}_r - \dot{s} \vec{t_r}^T \vec{n}_r + l \vec{n_r}^T \dot{\vec{n}}_r \end{aligned} \tag{3.27}$$

Combined with Equation 3.12, it can be seen:

$$\vec{n_r}' = \frac{d\vec{n}_r}{ds} = -k_r \vec{t}_r \tag{3.28}$$

$$\dot{\vec{n}}_r = \frac{ds}{dt}\frac{d\vec{n}_r}{ds} = -K_r \dot{s} \vec{t_r} \tag{3.29}$$

Based on Equations 3.28 and 3.29, Equation 3.27 can be expressed as:

$$\dot{l} = v_x \vec{t_x}^T \vec{n}_r - \dot{s} \vec{t_r}^T \vec{n}_r - l \dot{s} k_r \vec{n_r}^T \vec{t}_r \tag{3.30}$$

Since $\vec{n}_r$ and $\vec{t_r}$ are orthogonal vectors:

$$\vec{n_r}^T \vec{t}_r = \vec{t_r}^T \vec{n}_r = 0 \tag{3.31}$$

The Equation 3.27 can be further expressed as:

$$\begin{aligned} \dot{l} &= v_x \vec{t_x}^T \vec{n}_r \\ &= v_x \sin \Delta\theta \end{aligned} \tag{3.32}$$

Taking the derivative with respect to time $t$ on both sides of Equation 3.21:

$$v_x \vec{T}_x = \dot{s} \vec{T}_r + \dot{l} \vec{N}_r - \dot{s} k_r l \vec{T}_r \tag{3.33}$$

Multiply both sides of Equation 3.33 by $\vec{t_r}$:

$$\dot{s} = \frac{v_x \cos \Delta\theta}{1 - k_r l} \tag{3.34}$$

Since $l' = \frac{dl}{ds} = \frac{\dot{l}}{\dot{s}}$:

$$l' = (1 - k_r l) \tan \Delta\theta \tag{3.35}$$

According to Equation 3.35, $l''$ can be expressed as:

$$\begin{aligned} l'' &= \frac{dl'}{ds} \\ &= \frac{d\,(1 - k_r l)}{ds} \tan \Delta\theta + \frac{1 - k_r l}{\cos^2 \Delta\theta} \frac{d(\Delta\theta)}{ds} \end{aligned} \tag{3.36}$$

Where,

$$\frac{d\,(1 - k_r l)}{ds} = -\left(dk_r l + k_r l'\right) \tag{3.37}$$

$$\frac{d(\Delta\theta)}{ds} = \frac{d\theta_x}{ds} - \frac{d\theta_r}{ds} = \frac{1 - k_r l}{\cos\left(\theta_x - \theta_r\right)} \frac{d\theta_x}{ds_x} - \frac{d\theta_r}{ds} \tag{3.38}$$

$$k_r = \frac{d\theta_r}{ds} \tag{3.39}$$

$$k_x = \frac{d\theta_x}{ds_x} \tag{3.40}$$

Therefore, $l''$ can be further expressed as:

$$\begin{aligned} l'' = &-\left(dk_r l + k_r l'\right) \tan\left(\theta_x - \theta_r\right) + \\ &\frac{1 - k_r l}{\cos^2\left(\theta_x - \theta_r\right)} \left(k_x \frac{1 - k_r l}{\cos\left(\theta_x - \theta_r\right)} - k_r\right) \end{aligned} \tag{3.41}$$

According to Equation 3.34, we can obtain Equation 3.42

$$\begin{aligned} a_x &= \frac{dv_x}{dt} \\ &= \ddot{s} \frac{1 - k_r l}{\cos\left(\theta_x - \theta_r\right)} + \frac{\dot{s}^2}{\cos\left(\theta_x - \theta_r\right)} \left[l' \left(k_x \frac{1 - k_r l}{\cos\left(\theta_x - \theta_r\right)} - k_r\right) - \left(k_r' l + k_r l'\right)\right] \end{aligned} \tag{3.42}$$

Since $a_x$ is known, the $\ddot{s}$ can be expressed as:

$$\ddot{s} = \frac{a_x \cos\left(\theta_x - \theta_r\right) - \dot{s}^2 \left[l' \left(k_x \frac{1-k_r l}{\cos(\theta_x - \theta_r)} - k_r\right) - \left(k_r' l + k_r l'\right)\right]}{1 - k_r l} \tag{3.43}$$

In summary, the transformation relationship from Cartesian to Frenet frame is as follows:

$$\begin{cases} s = s_r \\ \dot{s} = \frac{v_x \cos \Delta\theta}{(1-k_r l)} \\ \ddot{s} = \frac{a_x \cos \Delta\theta - \dot{s}^2 \left[l' \left(k_x \frac{1-k_r l}{\cos \Delta\theta} - k_r\right) - \left(k_r' l + k_r l'\right)\right]}{1-k_r l} \\ l = \operatorname{sign}\left(\left(y_x - y_r\right)\cos\left(\theta_r\right) - \left(x_x - x_r\right)\sin\left(\theta_r\right)\right)\sqrt{\left(x_x - x_r\right)^2 + \left(y_x - y_r\right)^2} \\ \dot{l} = v_x \sin \Delta\theta \\ \ddot{l} = a_x \sin \Delta\theta \\ l' = \left(1 - k_r l\right)\tan \Delta\theta \\ l'' = -\left(k_r' l + k_r l'\right)\tan \Delta\theta + \frac{1-k_r l}{\cos^2 \Delta\theta}\left(k_p \frac{1-k_r l}{\cos \Delta\theta} - k_r\right) \end{cases} \tag{3.44}$$

### 3.3.3 Frenet Frame to Cartesian Frame

After obtaining the planned trajectory in the Frenet frame, it's usually necessary to transform the information of all trajectory points from the Frenet frame to the Cartesian frame before sending it to the control module for tracking. The transformation from the Frenet frame to the Cartesian frame can be referred to in Equation 3.44. We have omitted the derivation of this step [90]. In summary, the transformation relationship from Frenet to Cartesian frame is as follows:

$$\begin{cases} x_x = & x_r - d\sin\theta_r \\ y_x = & y_r + d\cos\theta_r \\ \theta_x = & \arctan\left(\frac{d'}{1-k_r d}\right) + \theta_r \in [-\pi, \pi] \\ v_x = & \sqrt{\left[\dot{s}\left(1 - k_r d\right)\right]^2 + \left(\dot{s} d'\right)^2} \\ a_x = & \ddot{s}\frac{1-k_r d}{\cos \Delta\theta} + \frac{\dot{s}^2}{\cos \Delta\theta}\cdot \\ & \left[d'\left(k_x \frac{1-k_r d}{\cos \Delta\theta)} - k_r\right) - \left(k_r' d + k_r d'\right)\right] \\ k_x = & \left(\left(d'' + \left(k_r' d + k_r d'\right)\tan \Delta\theta\right)\frac{\cos^2 \Delta\theta}{1-k_r d} + k_r\right) \end{cases} \tag{3.45}$$

## 3.4 Quadratic Programming

In trajectory planning, it is often necessary to find an optimal path to reach a given goal while satisfying a set of complex constraints. When dealing with planning problems, quadratic programming algorithms are often used to solve the following problems [91], [92]:

- Smooth trajectory: in practice, it is often desirable to generate smooth trajectories to reduce acceleration and curvature changes during the motion of a robot or vehicle. This improves comfort and stability. One way to achieve this is to minimize the curvature of the trajectory, typically in the form of a quadratic objective function.
- Dynamic constraints: trajectory planning must take into account dynamic constraints such as obstacle avoidance, adherence to traffic rules, and restrictions on a vehicle's maximum speed and acceleration. These constraints can be expressed using linear and quadratic inequalities, and QP is an effective method for handling these inequality constraints.
- Multi-Objective Optimisation: trajectory planning problems often involve balancing multiple objectives, such as minimizing travel time, minimizing travel distance, and minimizing curvature. This multi-objective optimization can be achieved by combining the weights of different objectives into a quadratic objective function.
- Continuous control variables: trajectory planning involves selecting control variables in continuous time and space, such as a vehicle's position, velocity, and acceleration. These continuous control variables lend themselves naturally to modeling using the QP algorithm.

A standard QP problem is defined as follows [93]:

$$\min \frac{1}{2} x^T H x + g^T x \tag{3.46}$$

$$\text{s.t. } a_i^T x = b_i, i = 1, 2, \ldots, m \tag{3.47}$$

where, $H$ is the Hessian matrix consisting of second-order derivatives; $g^t$ is the Jacobian matrix consisting of gradients.

The Lagrangian function for the standard QP problem is:

$$L(x, \lambda) = \frac{1}{2} x^T H x + g^T x + \sum_{i=1}^{m} \lambda_i \left( \alpha_i^T x - b_i \right) \tag{3.48}$$

The Karush-Kuhn-Tucker (KKT) condition is satisfied as shown in Equation 3.49.

$$\begin{cases} \frac{\partial L}{\partial x} = Hx + g + \sum_{i=1}^{m} \lambda_i \alpha_i = 0 \\ \\ \frac{\partial L}{\partial \lambda_i} = \alpha_i^T x - b_i = 0, i = 1, 2, \ldots, m \end{cases} \tag{3.49}$$

The matrix form of Equation 3.49 is shown in Equations 3.50 and 3.51.

$$F(x, \lambda) = \begin{pmatrix} Hx + g + \alpha \\ \alpha^T x - b \end{pmatrix} = 0 \tag{3.50}$$

$$\lambda = \begin{bmatrix} \lambda_1 & 0 & \ldots & 0 \\ 0 & \lambda_2 & \ldots & 0 \\ \ldots & \ldots & \ldots & \ldots \\ 0 & 0 & \ldots & \lambda_m \end{bmatrix}, \quad a^T = \begin{pmatrix} {a_1}^T \\ {a_2}^T \\ \ldots \\ {a_m}^T \end{pmatrix}, \quad b = \begin{pmatrix} b_1 \\ b_2 \\ \ldots \\ b_m \end{pmatrix} \tag{3.51}$$

# 4

# Multi-Objective Optimization Trajectory Planning on Urban Road

## 4.1 Background

Urban road scenarios are the main scenarios of daily traffic. These scenarios include straight roads, curved roads, intersections, and U-shaped roads, which are the main components of urban road networks [94]. The variety and complexity of the urban road scenarios reflect not only the reality of the urban road system but also the various challenges in the application of autonomous driving technologies [95]. For example, on straight roads, vehicles may need to maintain a stable straight line, while on curved roads, they may need to adapt to frequent turns. Therefore, the diversity of urban road scenarios places higher demands on path planning algorithms, which need to have good adaptability and robustness to adapt to different urban working conditions to ensure safe and efficient driving.

The primary criterion in the design of trajectory planning algorithms is to ensure the safety of the planned trajectories. However, it is essential to recognize that even in the absence of collisions, there might be underlying safety disparities among different collision-free candidate paths [96]. These differences arise due to factors such as the relative distances and spatial configurations between the paths and obstacles. Therefore, relying solely on collision detection is insufficient to guarantee overall safety. For instance, consider two collision-free paths, one of which is closer to a static obstacle while the other is further away [97]. Although neither path results in a collision, the path closer to the obstacle clearly presents a greater potential risk. In emergencies, vehicles need to have enough time and space to react

to unforeseen circumstances. If a path is too close to an obstacle, the vehicle may be constrained during emergency braking or evasion, limiting its ability to fully exploit its inherent safety. Therefore, when selecting the optimal path, it is imperative to assess the safety of collision-free candidate paths to ensure that autonomous vehicles have sufficient margin to avoid potential hazards in different situations, thereby increasing overall safety [98].

However, current trajectory planning algorithms, even when considering trajectory safety, often overlook the dimensions of obstacles when calculating path safety. In reality, obstacles are not simple points; they have unique dimensions and shapes. Consequently, the same path facing obstacles of different sizes at the same location surely has different safety values. Although the distances to the centers of these obstacles are the same, the distances to their edges vary. Therefore, larger obstacles evidently entail greater potential risks because vehicles are closer to them, increasing the likelihood of dangerous accidents in sudden emergencies.

In summary, in trajectory planning, we need to consider not only the safety of the planned trajectory but also the sizes of the obstacles when evaluating the safety of the trajectory. This is the only way to ensure that trajectory planning algorithms can effectively handle obstacles of different sizes and generate the safest possible paths.

## 4.2 Trajectory Planning Framework Overview

Within a Cartesian coordinate system, the interdependence of a vehicle's lateral and longitudinal motions makes motion calculations complex and computationally demanding. Consequently, the adoption of the Frenet frame becomes essential to disentangle these intertwined aspects of motion. This transformation simplifies the analysis of the vehicle's motion dynamics.

Furthermore, when considering the complex and non-convex nature of the vehicle's driving environment, obtaining a direct optimal solution becomes a formidable challenge. To address this, the non-convex problem is tactically transformed into a convex one by discretizing the motion space, thus creating a more tractable and computationally efficient model.

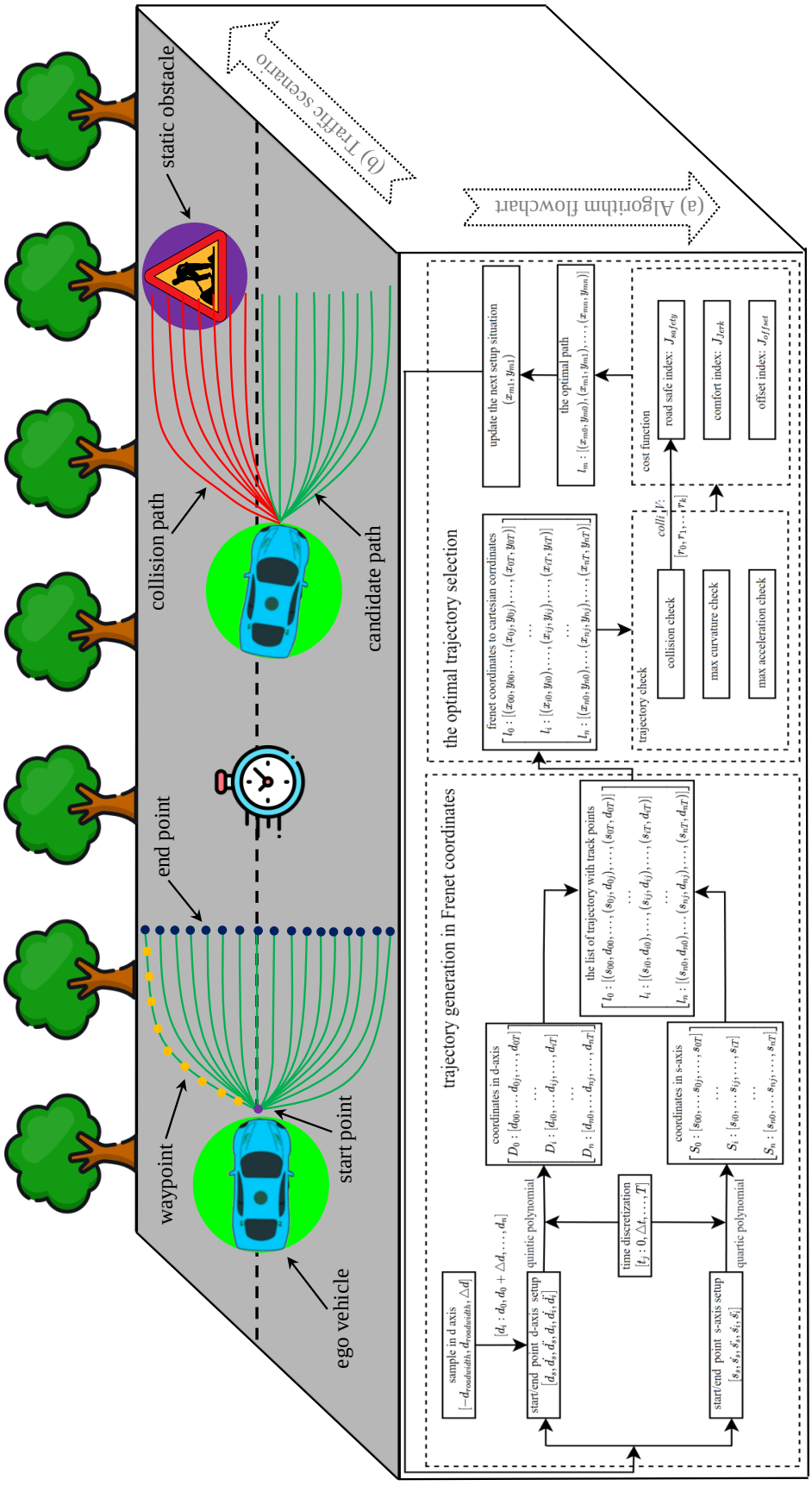



**Fig. 4.1:** Proposed algorithm framework. (a) the flowchart of the algorithm; (b) the illustration of the proposed algorithm in a real traffic scenario.

Subsequently, we proceed to compute the optimal path by evaluating a carefully constructed cost function along with appropriate constraints. This step ensures that the chosen trajectory is not only collision-free but also optimized in terms of various criteria such as ride comfort, safety, smoothness, etc.

The framework of our proposed algorithm, as shown in Figure 4.1, includes two crucial steps: trajectory generation within the Frenet frame and the judicious selection of the optimal trajectory. This structured approach not only simplifies the computation of vehicle motion but also navigates the complexity of the driving environment, enabling safer and more efficient autonomous navigation.

During the phase focused on generating trajectories within the Frenet frame, the principal procedures unfold in the following sections.

### 4.2.1 Spatial Discretization

The road width is parameterized by discrete intervals of $\Delta d$ along the $l$ axis. The upper bound of the road width is denoted by $d_{\text{roadwidth}}$, while the lower bound is denoted by $-d_{\text{roadwidth}}$. This discretization and boundary definition facilitates the accurate modeling of road width for subsequent analysis and trajectory planning.

### 4.2.2 Start/End Status Initialization

The states of the initial and final trajectory points have a significant influence on path generation. In the context of the Frenet frame, which effectively separates vehicle motion into lateral and longitudinal components, we denote the initial and final states along the $l$ axis as $\left[l_{\text{s}}, \dot{l}_{\text{s}}, \dot{l}_{\text{s}}, l_i, \dot{l}_i, \dot{l}_i\right]$. Where $l_{\text{s}}$ is the position coordinate of the start point of the trajectory along the $l$ axis and $l_i$ is the position coordinate of the endpoint of the trajectory along the $l$ axis. In this research, it is assumed that the autonomous vehicle is operating under adaptive cruise control, thereby maintaining the desired forward speed. Consequently, the initial and final states along the $s$ axis are expressed as

$[s_s, \dot{s}_s, \dot{s}_s, \dot{s}_i, \dot{s}_i]$, where $s_s$ is the position coordinate of the start trajectory point along the $s$ axis, and $s_i$ is the position coordinate of the end trajectory point along the $s$ axis. These states are consistent with the driving requirements and will serve as the basis for subsequent trajectory planning.

### 4.2.3 Trajectory Generation

The establishment of the vehicle's trajectory, both in the lateral ($l$-axis) and longitudinal ($s$-axis) directions, is a meticulous process. It begins with defining the initial and final states for each axis in Section 4.2.2, symbolized as $\left[l_s, \dot{l}_s, \ddot{l}_s, l_i, \dot{l}_i, \ddot{l}_i\right]$ for the $l$-axis, and $[s_s, \dot{s}_s, \ddot{s}_s, \dot{s}_i, \ddot{s}_i]$ for the $s$-axis.

The trajectory planning process unfolds in two stages, each taking into account the respective polynomial representation. A quintic polynomial is used to describe the lateral ($l$-axis) motion, while a quartic polynomial is used for the longitudinal ($s$-axis) motion description. To ensure a comprehensive trajectory plan, the planning cycle $T$ is discretized into smaller time intervals of $\Delta t$. This discretization produces a time series represented as $[0, \Delta t, ..., T]$. These time values are then inserted into the trajectory equations for both the $l$ axis and the $s$ axis. This process yields the coordinates of the trajectory points at each time step for both axes.

The computed $l$ and $s$ axis coordinates are combined to produce the final planned trajectories that can effectively guide the vehicle. For a more detailed exploration of this methodology, readers are encouraged to refer to Section 4.3 and Section 4.4. This systematic approach ensures the seamless integration of lateral and longitudinal trajectory planning within the autonomous driving system.

In the optimal trajectory selection phase, the following key steps are meticulously carried out to ensure a comprehensive evaluation of trajectory candidates, helping to select the most suitable path for the vehicle in the given road context.

### 4.2.4 Trajectory Point Coordinate Transformation

To ensure the accuracy of the collision check and control module, it is necessary to transform the trajectory set from the Frenet frame to the Cartesian frame. This transformation is essential since, in tracking tasks, we typically rely on the trajectory information in the Cartesian frame for accurate position tracking and control operations. Furthermore, by converting the trajectory set from the Frenet frame to the Cartesian frame, we can perform a more precise collision check on the generated trajectories, thereby ensuring the safety of the planned trajectories.

Based on Equation 3.45, the trajectories generated in the Frenet frame are then transformed into the Cartesian frame.

### 4.2.5 Trajectory Collision Check

A trajectory is fundamentally composed of multiple trajectory points, each of which contains spatio-temporal information such as location, curvature, heading, velocity, and acceleration, among others. In this study, the trajectory evaluation process focuses exclusively on examining the collision, acceleration, and curvature for each trajectory point along the path.

The trajectory equations and trajectory points are obtained in the previous sections, specifically in Section 4.2.2 and Section 4.2.3. At this stage, a virtual ego vehicle is virtually positioned at each trajectory point, and collision detection is performed by assessing whether there is any overlap between the boundaries of the virtual ego vehicle and those of the obstacles, as shown in Figure 4.1(b).

Taking into account the constraints imposed by the kinematics and dynamics of the vehicle, the acceleration and curvature of each trajectory point are rigorously examined. Trajectories are eliminated from consideration if any trajectory point within them fails to meet the criteria of collision avoidance, permissible acceleration, and steering constraints.

Full details of the specific operations and derivation processes can be found in section 4.5. This multi-stage approach results in the generation of candidate trajectories that are both collision-free and satisfy the specified constraints.

### 4.2.6 Optimal Trajectory Selection

The loss for each trajectory is computed, and the trajectory exhibiting the lowest total loss is selected as the optimal trajectory. The cost function encompasses three distinct components: comfort, trajectory safety, and trajectory anti-deviation. A comprehensive description of these three metrics is provided in Section 4.6.

### 4.2.7 The Start Point Status Update

The planning frequency is represented as $1/\Delta t$. As the ego vehicle moves forward, its position changes continuously. Consequently, our trajectory planning method uses the current vehicle position as a central reference for generating the subsequent trajectory.

We denote the current vehicle position as $(x_0, y_{m0})$ and proceed to compute the trajectory point $(x_1, y_{m1})$ corresponding to a time interval of $\Delta t$ seconds into the future. This newly calculated point becomes the starting point for the subsequent iteration of trajectory planning. This approach ensures that the trajectory planning process remains synchronized with the dynamic motion of the ego vehicle, allowing for real-time adjustments based on its evolving position. Detailed insights into this process are provided in the following sections of this paper.

## 4.3 Lateral Motion Trajectory Planning

Lateral motion planning includes critical objectives such as obstacle avoidance and lane changes. According to Equation 4.1, the lateral motion trajectory $l(t)$ is aptly represented by the utilization of a quintic polynomial curve.

This choice is supported by the information provided in section 4.2.2, which allows us to define the initial and final states of the lateral motion trajectory as $\left[l_{\mathrm{s}}, \dot{l}_{\mathrm{s}}, \ddot{l}_{\mathrm{s}}, l_i, \dot{l}_i, \dot{l}_i\right]$. These states include the requirements for initial and final positions, velocities, and accelerations. This means that there are six equality constraints, which correspond to the six coefficients needed to describe a quintic polynomial. Consequently, these constraints form a full-rank matrix that uniquely determines the coefficients of the quintic polynomial curve [99]. This mathematical model selection not only ensures a well-defined trajectory but also accommodates complex trajectory generation tasks, providing mathematical consistency and solvability. This makes the model suitable for critical lateral motion tasks such as obstacle avoidance and lane changes.

$$l(t) = c_{l0} + c_{l1}t + c_{l2}t^2 + c_{l3}t^3 + c_{l4}t^4 + c_{l5}t^5 \tag{4.1}$$

where $c_{l0}$, $c_{l1}$, $c_{l2}$, $c_{l3}$, $c_{l4}$, and $c_{l5}$ are the coefficients of a quintic polynomial.

Taking the first-order derivative of both sides of Equation 4.1 with respect to time $t$, we obtain the following Equation:

$$\dot{l}(t) = c_{l1} + 2c_{l2}t + 3c_{l3}t^2 + 4c_{l4}t^3 + 5c_{l5}t^4 \tag{4.2}$$

The second-order derivative of Equation 4.1 with respect to time $t$ is denoted as:

$$\ddot{l}(t) = 2c_{l2} + 6c_{l3}t + 12c_{l4}t^2 + 20c_{l5}t^3 \tag{4.3}$$

The given initial and final conditions at the times $t_{\mathrm{s}}$ and $t_{\mathrm{e}}$ are incorporated into Equation 4.1 through Equation 4.3:

$$\begin{pmatrix} l\left(t_{\mathrm{s}}\right) \\ \dot{l}\left(t_{\mathrm{s}}\right) \\ \ddot{l}\left(t_{\mathrm{s}}\right) \\ l\left(t_{\mathrm{e}}\right) \\ \dot{l}\left(t_{\mathrm{e}}\right) \\ \ddot{l}\left(t_{\mathrm{e}}\right) \end{pmatrix} = \begin{pmatrix} 1 & t_{\mathrm{s}} & {t_{\mathrm{s}}}^2 & {t_{\mathrm{s}}}^3 & {t_{\mathrm{s}}}^4 & {t_{\mathrm{s}}}^5 \\ 0 & 1 & 2t_{\mathrm{s}} & 3{t_{\mathrm{s}}}^2 & 4{t_{\mathrm{s}}}^3 & 5{t_{\mathrm{s}}}^4 \\ 0 & 0 & 2 & 6t_{\mathrm{s}} & 12{t_{\mathrm{s}}}^2 & 20{t_{\mathrm{s}}}^3 \\ 1 & t_{\mathrm{e}} & {t_{\mathrm{e}}}^2 & {t_{\mathrm{e}}}^3 & {t_{\mathrm{e}}}^4 & {t_{\mathrm{e}}}^5 \\ 0 & 1 & 2t_{\mathrm{e}} & 3{t_{\mathrm{e}}}^2 & 4{t_{\mathrm{e}}}^3 & 5{t_{\mathrm{e}}}^4 \\ 0 & 0 & 2 & 6t_{\mathrm{e}} & 12{t_{\mathrm{e}}}^2 & 20{t_{\mathrm{e}}}^3 \end{pmatrix} \begin{pmatrix} c_{l0} \\ c_{l1} \\ c_{l2} \\ c_{l3} \\ c_{l4} \\ c_{l5} \end{pmatrix} \tag{4.4}$$

Equation 4.5 is derived through a modification of Equation 4.4, and the coefficient matrix can be computed as follows:

$$\begin{pmatrix} c_{l0} \\ c_{l1} \\ c_{l2} \\ c_{l3} \\ c_{l4} \\ c_{l5} \end{pmatrix} = \begin{pmatrix} 1 & t_{\mathrm{s}} & {t_{\mathrm{s}}}^2 & {t_{\mathrm{s}}}^3 & {t_{\mathrm{s}}}^4 & {t_{\mathrm{s}}}^5 \\ 0 & 1 & 2t_{\mathrm{s}} & 3{t_{\mathrm{s}}}^2 & 4{t_{\mathrm{s}}}^3 & 5{t_{\mathrm{s}}}^4 \\ 0 & 0 & 2 & 6t_{\mathrm{s}} & 12{t_{\mathrm{s}}}^2 & 20{t_{\mathrm{s}}}^3 \\ 1 & t_{\mathrm{e}} & {t_{\mathrm{e}}}^2 & {t_{\mathrm{e}}}^3 & {t_{\mathrm{e}}}^4 & {t_{\mathrm{e}}}^5 \\ 0 & 1 & 2t_{\mathrm{e}} & 3{t_{\mathrm{e}}}^2 & 4{t_{\mathrm{e}}}^3 & 5{t_{\mathrm{e}}}^4 \\ 0 & 0 & 2 & 6t_{\mathrm{e}} & 12{t_{\mathrm{e}}}^2 & 20{t_{\mathrm{e}}}^3 \end{pmatrix}^{-1} \begin{pmatrix} l\left(t_{\mathrm{s}}\right) \\ \dot{l}\left(t_{\mathrm{s}}\right) \\ \ddot{l}\left(t_{\mathrm{s}}\right) \\ l\left(t_{\mathrm{e}}\right) \\ \dot{l}\left(t_{\mathrm{e}}\right) \\ \ddot{l}\left(t_{\mathrm{e}}\right) \end{pmatrix} \tag{4.5}$$

## 4.4 Longitudinal Motion Trajectory Planning

According to Section 4.2.2, the initial and final states along the $s$ axis are $[s_{\mathrm{s}}, \dot{s}_{\mathrm{s}}, \dot{s}_{\mathrm{s}}, \dot{s}_i, \dot{s}_i]$. Hence, there are five equality constraints. Therefore, for modeling the Longitudinal motion trajectory, a fourth-order polynomial can be selected.

$$s(t) = c_{s0} + c_{s1}t + c_{s2}t^2 + c_{s3}t^3 + c_{s4}t^4 \tag{4.6}$$

where $c_{s0}$, $c_{s1}$, $c_{s2}$, $c_{s3}$, and $c_{s4}$ are the coefficients of a quartic polynomial curve.

The first-order derivative of Equation 4.6 with respect to time $t$ is denoted as:

$$\dot{s}(t) = c_{s1} + 2c_{s2}t + 3c_{s3}t^2 + 4c_{s4}t^3 \tag{4.7}$$

The second-order derivative of Equation 4.6 with respect to time $t$ is denoted as:

$$\ddot{s}(t) = 2c_{s2} + 6c_{s3}t + 12c_{s4}t^2 \tag{4.8}$$

Equation 4.9 can be derived based on Equations 4.6 to 4.8, considering the specified initial and final positions at times $t_\mathrm{s}$ and $t_\mathrm{e}$:

$$\begin{pmatrix} s\left(t_\mathrm{s}\right) \\ \dot{s}\left(t_\mathrm{s}\right) \\ \ddot{s}\left(t_\mathrm{s}\right) \\ \dot{s}\left(t_\mathrm{e}\right) \\ \ddot{s}\left(t_\mathrm{e}\right) \end{pmatrix} = \begin{pmatrix} 1 & t_\mathrm{s} & {t_\mathrm{s}}^2 & {t_\mathrm{s}}^3 & {t_\mathrm{s}}^4 \\ 0 & 1 & 2t_\mathrm{s} & 3{t_\mathrm{s}}^2 & 4{t_\mathrm{s}}^3 \\ 0 & 0 & 2 & 6t_\mathrm{s} & 12{t_\mathrm{s}}^2 \\ 1 & t_\mathrm{e} & {t_\mathrm{e}}^2 & {t_\mathrm{e}}^3 & {t_\mathrm{e}}^4 \\ 0 & 1 & 2t_\mathrm{e} & 3{t_\mathrm{e}}^2 & 4{t_\mathrm{e}}^3 \\ 0 & 0 & 2 & 6t_\mathrm{e} & 12{t_\mathrm{e}}^2 \end{pmatrix} \begin{pmatrix} c_{s0} \\ c_{s1} \\ c_{s2} \\ c_{s3} \\ c_{s4} \end{pmatrix} \tag{4.9}$$

Ultimately, the quartic polynomial coefficients can be determined by performing a transformation on Equation 4.9:

$$\begin{pmatrix} c_{s0} \\ c_{s1} \\ c_{s2} \\ c_{s3} \\ c_{s4} \end{pmatrix} = \begin{pmatrix} 1 & t_\mathrm{s} & {t_\mathrm{s}}^2 & {t_\mathrm{s}}^3 & {t_\mathrm{s}}^4 \\ 0 & 1 & 2t_\mathrm{s} & 3{t_\mathrm{s}}^2 & 4{t_\mathrm{s}}^3 \\ 0 & 0 & 2 & 6t_\mathrm{s} & 12{t_\mathrm{s}}^2 \\ 0 & 1 & 2t_\mathrm{e} & 3{t_\mathrm{e}}^2 & 4{t_\mathrm{e}}^3 \\ 0 & 0 & 2 & 6t_\mathrm{e} & 12{t_\mathrm{e}}^2 \end{pmatrix}^{-1} \begin{pmatrix} s\left(t_\mathrm{s}\right) \\ \dot{s}\left(t_\mathrm{s}\right) \\ \ddot{s}\left(t_\mathrm{s}\right) \\ \dot{s}\left(t_\mathrm{e}\right) \\ \ddot{s}\left(t_\mathrm{e}\right) \end{pmatrix} \tag{4.10}$$

## 4.5 Candidate Path Generation

Figure 4.1 illustrates the collision potential of the generated trajectories and the constraints on the vehicle's motion and dynamic characteristics. To improve the response time of the system, trajectories that do not meet the constraints are subjected to a trajectory verification process. The remaining trajectories are then presented as candidate paths for the subsequent module to select the optimal path. The main components of the verification include collision, curvature, and acceleration assessments.

### 4.5.1 Collision Check

Circular shapes are employed as bounding boxes for collision assessments. The criteria for successfully passing the collision assessment are as outlined below:

$$(x_{ij} - x_{\text{ob}})^2 + (y_{ij} - y_{\text{ob}})^2 < (r_{\text{ego}} + r_{\text{ob}})^2 \tag{4.11}$$

where, $(x_{\text{ob}}, y_{\text{ob}})$ represents the coordinates of a specific obstacle, while $(x_{ij}, y_{ij})$ denote the coordinates of trajectory point $j$ on trajectory $i$. $r_{\text{ego}}$ corresponds to the radius of the circular bounding box for the ego vehicle, and $r_{\text{ob}}$ pertains to the radius of the circular bounding box for the obstacle.

### 4.5.2 Curvature Check

The curvature value at each point along the trajectory has a direct effect on the vehicle's steering performance and comfort. Performing a curvature assessment allows the curvature characteristics of the trajectory to be evaluated to ensure that the vehicle can follow the planned path without exceeding its handling limits, thus avoiding unstable turns or dangerous driving situations.

$$\kappa_x[ij] \leqslant \kappa_{\max}, 0 \leqslant j \leqslant N \tag{4.12}$$

where $\kappa_{\max}$ is the maximum allowed curvature, and $N$ is the number of trajectory points.

### 4.5.3 Acceleration Check

Acceleration is a crucial parameter in describing how a vehicle changes speed along a trajectory and it has a direct impact on the vehicle's acceleration and deceleration behavior. By performing an acceleration assessment, the acceleration characteristics at each point along the trajectory can be evaluated to ensure that the vehicle does not experience excessive acceleration or deceleration as it follows the trajectory, thereby providing a smoother and more comfortable driving experience. Furthermore, taking into account the vehicle's dynamic performance, acceleration assessment also helps to prevent

uncontrollable acceleration or braking situations during the execution of the planned path, thereby increasing the vehicle's safety and feasibility in various driving scenarios.

$$a_x[ij] \leqslant a_{\max}, 0 \leqslant j \leqslant N \tag{4.13}$$

where $a_{\max}$ is the maximum allowed acceleration.

## 4.6 Cost Function Definition

After the trajectory evaluation, a set of candidate trajectories is generated. However, the number of candidate trajectories remains substantial, so it is necessary to select a single trajectory for execution. To achieve this, we formulate a cost function that evaluates each candidate trajectory and identifies the one with the lowest cost as the most optimal choice. The design of the cost function in this paper is based on three criteria: trajectory comfort, safety, and deviation from the road center line. The cost function, denoted as $J[i]$, is defined in Equation 4.14:

$$J[i] = w_0 J_{\text{jerk}}[i] + w_1 J_{\text{safety}}[i] + w_2 J_{\text{offset}}[i] \tag{4.14}$$

$J_{\text{jerk}}$, $J_{\text{safety}}$, and $J_{\text{offset}}$ represent the cost functions associated with jerk, trajectory safety, and deviation from the road center line, respectively. Additionally, $w_0$, $w_1$, and $w_2$ denote the weighting factors applied to these cost functions, influencing the vehicle's driving behavior and style.

### 4.6.1 Jerk Indicator

Jerk is a critical metric for evaluating the quality of planned trajectories. Typically, the derivative of acceleration is employed to quantify jerk. As the Frenet frame decouples lateral and longitudinal motion, the jerk value can be formulated by aggregating the derivatives of lateral and longitudinal accelerations.

$$J_{\text{jerk}}[i](t) = \int_{t_0}^{t_1} \dddot{l}^2(t)\mathrm{d}t + \int_{t_0}^{t_1} \dddot{s}^2(t)\mathrm{d}t \tag{4.15}$$

### 4.6.2 Safety Indicator

While all candidate trajectories are collision-free, their safety levels vary due to their proximity to obstacles. It is evident that trajectories situated at a greater distance from obstacles tend to exhibit higher levels of safety. Wei et al. [100] introduces an approach for evaluating the safety of candidate trajectories based on their proximity to obstacles. However, this method does not fully consider the influence of obstacle size on trajectory safety. When the distance between the candidate trajectory and the center of an obstacle remains constant, the actual distance from the candidate trajectory to the obstacle's edge can differ. As a result, the safety of a candidate trajectory should be rated higher when navigating smaller obstacles, leading to a reduced safety cost.

In this study, we have drawn inspiration from the Gauss-Laplace operator and the concept of convolution in the field of computer vision to create a convolution kernel denoted as $f(x)$. We utilize this kernel to perform a convolution operation on the collision value, $colli\_V$, to compute the safety cost associated with each trajectory.

$$J_{\text{safety}}[i] = f * colli\_V[i] \tag{4.16}$$

The definition of the convolution kernel $f(x)$ is as follows:

$$f(x) = g(x) + \min(|g(x)|) \tag{4.17}$$

where $g(x)$ is the variation of the Gauss Laplace operator:

$$g(x) = \frac{1}{2\pi\sigma} \cdot e^{-\frac{(x-u)^2}{2\sigma^2}} \cdot \frac{\sigma^2 - x^2}{\sigma^4} \tag{4.18}$$

and $\sigma$ and $u$ are the standard deviations and mean values, respectively.

Given that any plane geometry can be enclosed by its circumcircle, our approach in this research leverages the circumcircle as a mathematical model for characterizing obstacles. The collision value, denoted as $colli\_V$, is determined by the size of the largest obstacle encountered along the trajectory.

$$colli\_V[i] = \begin{cases} r_{\text{ob}}[i], & \text{collision happen} \\ 0, & \text{collision free} \end{cases} \tag{4.19}$$

$r_{\text{ob}}$ is the radius of the obstacle-circumscribed circle.

Figure 4.2 illustrates how the collision value signifies the presence of collisions on the generated trajectories with various obstacle sizes. The safety cost of each candidate trajectory can be effectively computed based on the size and spatial distribution of the obstacles. Proximity to larger obstacles results in higher safety costs and lower trajectory safety. Likewise, trajectories that come closer to densely clustered obstacle formations incur higher safety costs, indicating reduced safety levels.

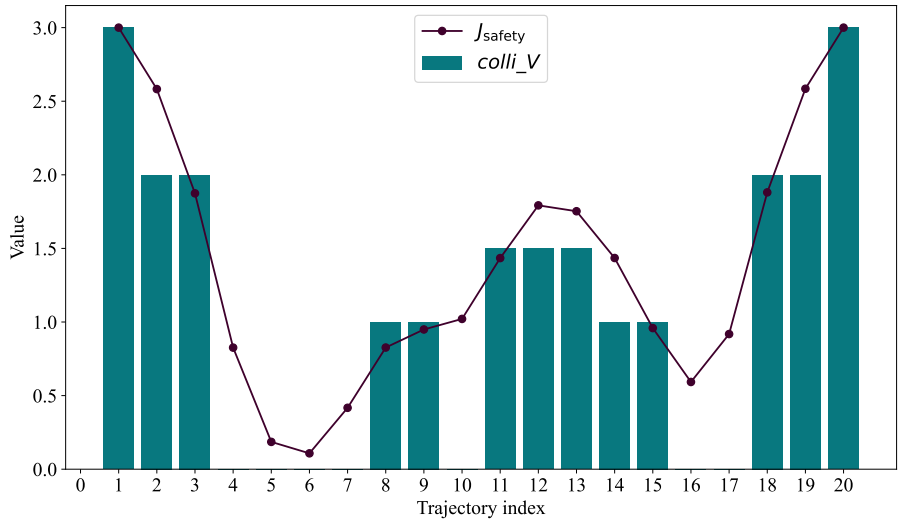



**Fig. 4.2:** Safety cost of each candidate trajectory

### 4.6.3 Anti-deviation Indicator

When driving a vehicle, deviation from the central axis of the road can result in a limited manoeuvring range when encountering obstacles, requiring sig-

nificant steering input to avoid them. This behavior not only inconveniences other road participants but also increases the potential for misjudgment, thereby increasing the risk of traffic accidents. In addition, it can often trigger lane departure warnings, leading to functional conflicts between the various modules within the vehicle's control system. Therefore, when selecting the optimal trajectory, it is imperative to ensure that the chosen path closely follows the center line of the road. In this paper, we introduce a cost function tailored to compute the integral of the square of the lateral displacement $l$ along the proposed trajectory to meet this essential requirement.

$$J_{\text{offset}}[i] = \int_{s_0}^{s_1} l^2(s)\mathrm{d}s / \int_{t_0}^{t_1} s^2(t)\mathrm{d}t \tag{4.20}$$

Given the discretization of the planning time, we can rephrase Equation 4.20 as follows:

$$J_{\text{offset}}[i] = \sum_{j=0}^{N} l_{ij}^2 / \sum_{j=0}^{N} s_{ij}^2 \tag{4.21}$$

where, $N$ represents the count of trajectory points within trajectory $i$.

## 4.7 Experiment and Discussion

It is common practice in path planning research to assume the availability of mapping, localization, and detection data, often without a detailed exploration of the functionalities of the front-end modules within autonomous driving systems. This study relies on traffic scenarios found in existing public perception datasets [101], [102], and the simulation settings presented in [103]. Specifically, within the simulation environment, various road configurations such as straight roads, curved roads, intersections, and 'U'-shaped roads are implemented using Python. In addition, several static obstacles of different sizes are placed along these roads. The experiments conducted in this study are divided into two main sections: the first part investigates the influence of different cost functions on trajectory generation, and the second

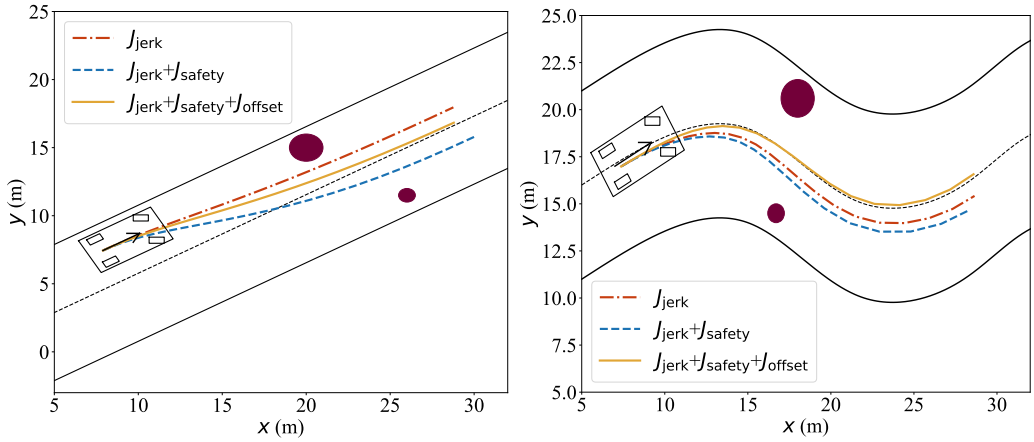
part provides a comparative analysis of the method proposed in this paper with those presented in [100]. Detailed simulation parameters are given in Table 4.1.

**Tab. 4.1:** Experiment Parameters.

| | Parameter Name | Value |
|---|---|---|
| $r_{\text{car}}$ (m) | Vehicle collision radius | 1 |
| $d_{\text{roadwidth}}$ (m) | Road width | 10 |
| $w_0$ | Weight of $J_{\text{jerk}}$ | 0.4 |
| $w_1$ | Weight of $J_{\text{safety}}$ | 0.3 |
| $w_2$ | Weight of $J_{\text{offset}}$ | 0.3 |
| $\Delta t$ (s) | Sampling time | 0.1 |
| $\Delta d$ (m) | Transverse sampling distance | 0.5 |
| $k_{\text{max}}$ ($\text{m}^{-1}$) | Maximum curvature | 0.5 |
| $a_{\text{max}}$ ($\text{m} \cdot \text{s}^{-2}$) | Maximum acceleration | 3 |
| $T_{\text{max}}$ (s) | Maximum planning period | 5 |
| $T_{\text{min}}$ (s) | Minimum planning period | 4 |
| $V_{\text{target}}$ ($\text{km} \cdot \text{h}^{-1}$) | Target speed | 50 |

### 4.7.1 Influence of Cost Function on Trajectory Selection

Figure 4.3 (a) and (b) illustrate how distinct cost functions yield different optimal trajectories on both straight and curved roads. This study examines the effects of three different cost functions on trajectory selection. When the cost function considers only jerk, the vehicle attempts to maintain its current trajectory while avoiding collisions. While this approach provides a comfortable ride, it may not provide an adequate level of safety and may bring the vehicle too close to obstacles. The second cost function takes into account both the $J_{\text{jerk}}$ and $J_{\text{safety}}$ metrics. As the safety metric in this study takes into account the size of the obstacle, it is evident that under the influence of this safety metric, the vehicle chooses an avoidance distance commensurate with the dimensions of the obstacle. Nevertheless, the resulting trajectory may still show deviations from the center line of the road after the obstacle has been avoided. The third cost function investigated in this study, which takes into account jerk, safety, and lateral deviation from the road center line, produces a smoother trajectory that ensures safe separation from obstacles and returns to the road center line after obstacle avoidance.




**(a)** On a straight road **(b)** On a curved road

**Fig. 4.3:** Influence of cost function on trajectory selection

### 4.7.2 Performance of Proposed Algorithm

To assess the feasibility and effectiveness of the local path planning algorithm proposed in this study, we performed simulations for both the algorithm presented in this paper and the one described in [100]. We then evaluated their performance in four different traffic scenarios, and the comparative results are summarised in Table 4.2. In the following, we refer to the method in [100] as the conventional method.

**Tab. 4.2:** Performance Comparison.

| Scenario | Algorithm | $J_{offset}$ | Decline Rate(%) | $J_{jerk}$ | Decline Rate(%) |
|---|---|---|---|---|---|
| Straight Road | proposed | 0.83 | 63.72% | 0.63 | 13.47% |
| | conventional | 2.29 | | 0.73 | |
| Curved Road | proposed | 0.69 | 13.86% | 0.6 | 32.19% |
| | conventional | 0.8 | | 0.9 | |
| Intersection | proposed | 0.18 | 44.36% | 0.06 | 59.36% |
| | conventional | 0.33 | | 0.16 | |
| U-shaped Road | proposed | 0.42 | 45.56% | 0.07 | 18.60% |
| | conventional | 0.77 | | 0.08 | |

### Straight road scenario

To validate the ability of the algorithm to continuously avoid obstacles, a scenario with a straight road and static obstacles was created in this study. As shown in Figure 4.4, both our approach and the conventional method successfully navigate around obstacles and reach their respective destinations safely. However, the global routing generated by the conventional method shows significant deviations, almost forming an s-shaped curve, and fails to realign with the center line of the road after obstacle avoidance. This can be attributed to the fact that the safety cost function used in the conventional method is insensitive to obstacle size, mechanically evaluating candidate trajectories based solely on their proximity to obstacles. As a result, this method tends to favor trajectories with significant distance from obstacles during obstacle avoidance, as there are no constraints on maintaining alignment with the center line of the road. In contrast, the approach presented in this paper incorporates three key metrics, namely $J_{\text{jerk}}$, $J_{\text{safety}}$, and $J_{\text{offset}}$, allowing it to safely navigate around obstacles and re-establish a connection to the road center line after avoidance. The global routing generated in this study has a relatively smoother trajectory.

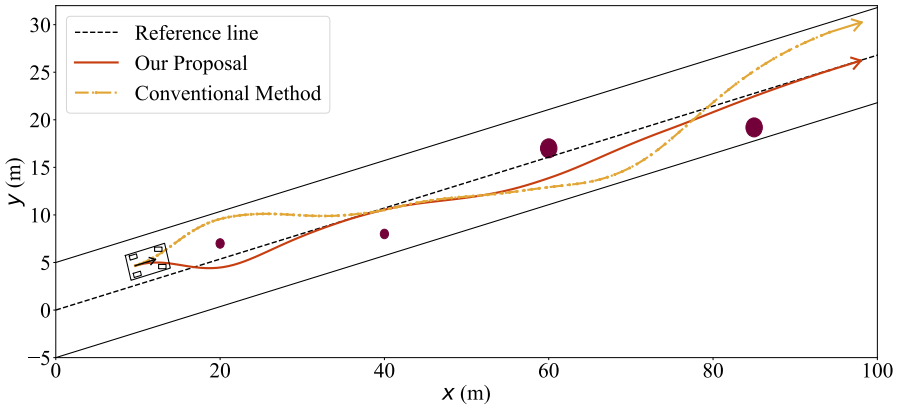



**Fig. 4.4:** Comparison of Global Routing on a Straight Road

Figure 4.5 provides a more distinct visualization of the vehicle's deviation from the road's center line. The method introduced in this paper outperforms the approach from the conventional method by effectively realigning the vehicle with the road's center line following obstacle avoidance. This achievement mitigates risks and hazards associated with undertaking extensive detours to evade obstacles. Notably, the mean value of $J_{\text{offset}}$ diminishes by

63.72%. In Figure 4.6, it is evident that on a straight road, during the initial 20 seconds, the algorithm presented in this paper necessitates considerations of road center line offset, which leads to a slightly higher $J_{\text{jerk}}$ compared to the conventional method. However, in the subsequent obstacle avoidance phase, the $J_{\text{jerk}}$ substantially improves, resulting in a 13.47% reduction in the mean $J_{\text{jerk}}$ throughout the entire process.

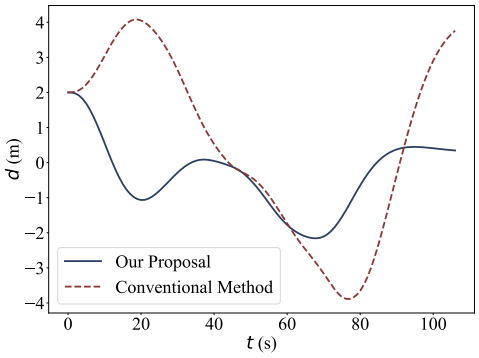



**Fig. 4.5:** Anti-deviation Comparison

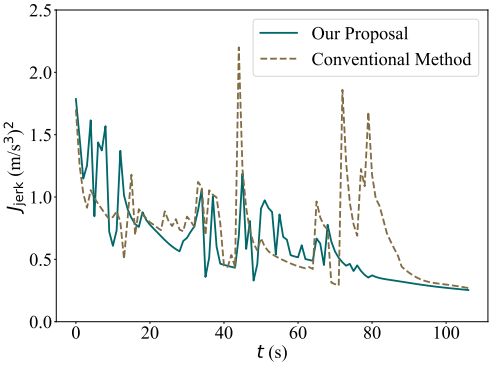



**Fig. 4.6:** Comfort Comparison

### Curvy Road Scenario

To assess the algorithm's performance in sustained curved road environments, an 'S'-shaped road scenario was intentionally created.

Figure 4.7 visually depicts the entire trajectory of the vehicle, starting from the starting point and ending at the destination on the S-shaped road. When navigating the curved road, the vehicle utilizes the cost function introduced in this paper, initiating obstacle avoidance at approximately 22 meters and successfully re-aligning with the road center line after obstacle avoidance, facilitated by the influence of $J_{\text{offset}}$. In contrast, the approach in the conventional method begins obstacle avoidance at approximately 25 meters. Despite the shorter longitudinal avoidance distance compared to the method proposed in this paper, the absence of $J_{\text{offset}}$ as a constraining factor results in the vehicle not returning to the center line of the road until 60 meters after obstacle avoidance, despite a constant approach to the center line of the road.

As depicted in Figure 4.8, between approximately 20 seconds and 40 seconds, the $J_{\text{offset}}$ in this study exhibits a larger value. This is attributed to our

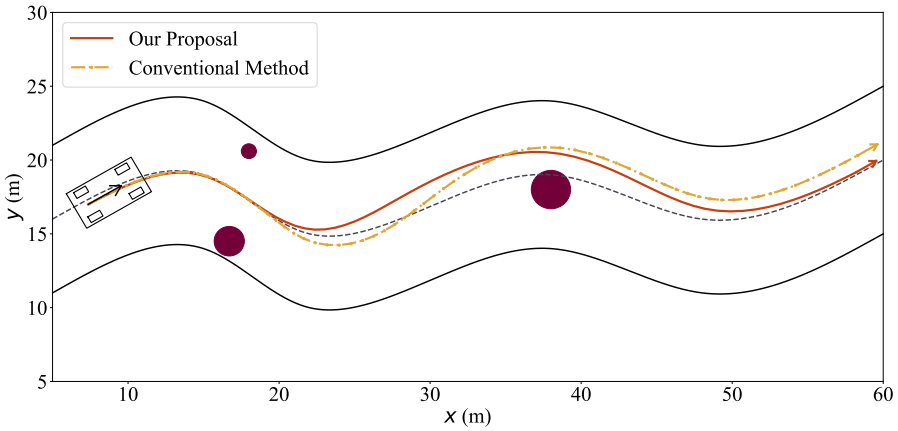



**Fig. 4.7:** Comparison of Global Routing on a Curvy Road

method initiating obstacle avoidance at around 22 meters, earlier than the conventional method, ensuring a timely return to the road's center line. Over the entire process, the mean value of $J_{\text{offset}}$ on the curved road decreases by 13.86%.

Figure 4.9 presents a comparison of $J_{\text{jerk}}$. On the curved road, the $J_{\text{jerk}}$ values throughout the process are smaller, with a mean $J_{\text{jerk}}$ reduction of 32.19%. This indicates that the algorithm introduced in this paper can offer superior driving comfort.

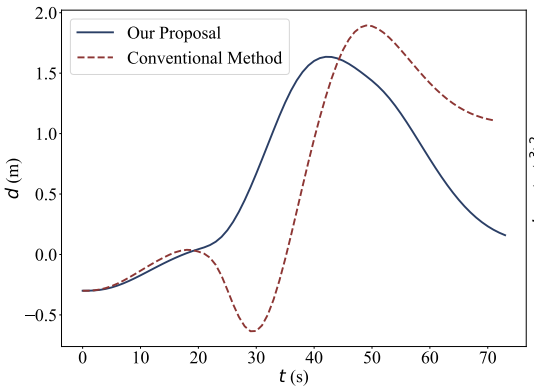



**Fig. 4.8:** Anti-deviation Comparison

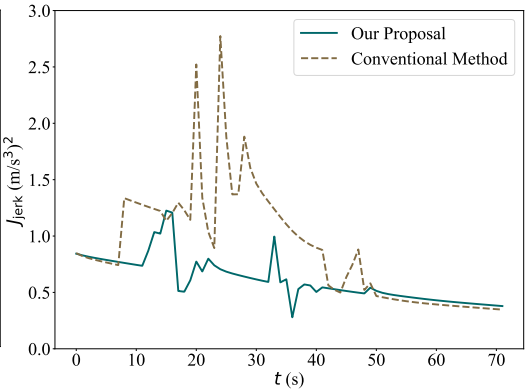



**Fig. 4.9:** Comfort Comparison

### Intersection scenario

Intersections represent ubiquitous traffic scenarios in real-world driving. In order to assess the efficacy of the algorithms, a simulated intersection traffic scenario was specifically designed.

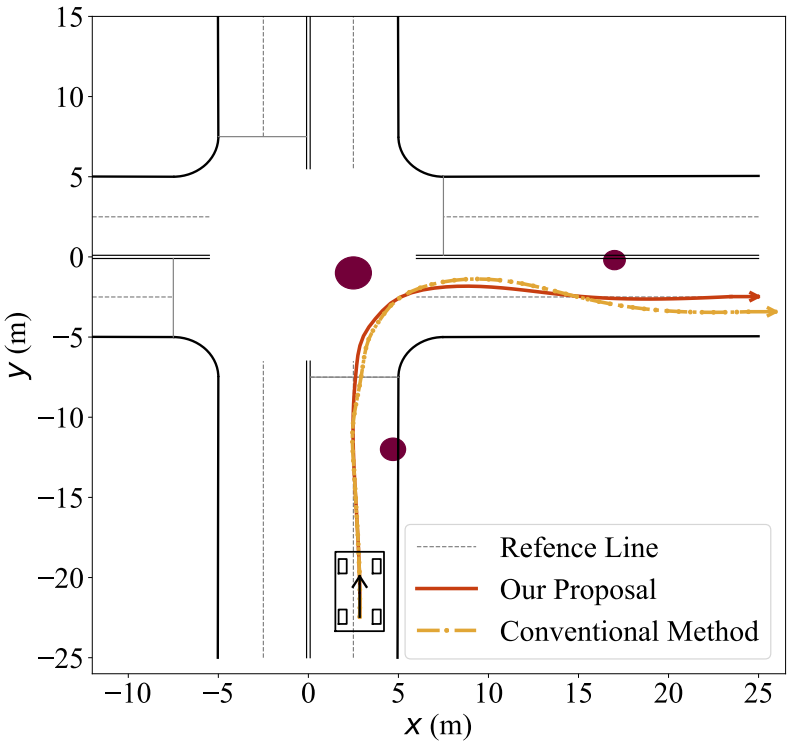



**Fig. 4.10:** Comparison of Global Routing in Intersection Scenario

Figure 4.10 illustrates the complete vehicle trajectory during the junction obstacle avoidance process. Both the algorithms proposed in this study and reference [100] show effectiveness in obstacle avoidance. However, our approach in this paper incorporates a cost function that considers the influence of obstacle size on road safety, facilitating skillful navigation while maintaining appropriate distances from obstacles of different dimensions. Furthermore, considering the deviation from the road center line, our algorithm proactively guides the path back to the road center line after avoiding obstacles.

In contrast, the conventional method relies solely on the distance between obstacles to assess the safety value of potential paths. Consequently, it tends to maintain a maximum distance from obstacles during obstacle avoidance,

resulting in less fluid and more erratic trajectories. Furthermore, due to the lack of consideration for maintaining alignment with the center line of the road, it often fails to realign with the center line of the road after obstacle avoidance and continues in its current direction influenced by $J_{\text{jerk}}$.

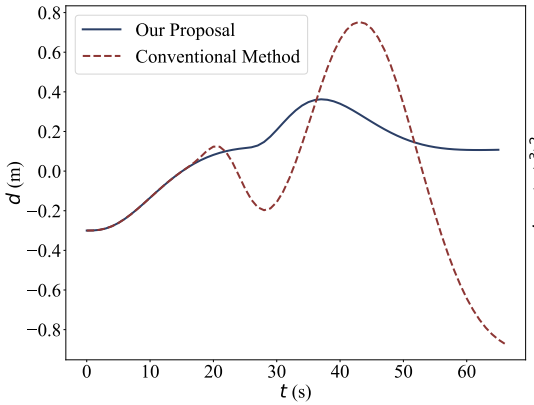



**Fig. 4.11:** Anti-deviation Comparison

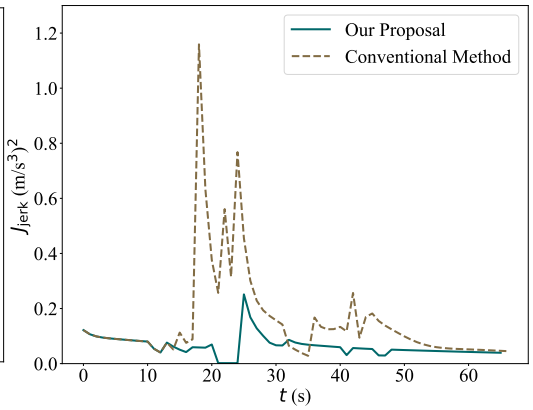



**Fig. 4.12:** Comfort Comparison

Figure 4.11 visualizes the deviation from the road center line over the entire trajectory. Between approximately seconds and 18 seconds, both algorithms show comparable deviations from the center line of the road. However, between 18 seconds and 35 seconds, the algorithm presented in this paper, influenced by $J_{\text{jerk}}$ and $J_{\text{offset}}$, attempts to maintain the current direction of travel while maintaining a safe distance from the second obstacle encountered.

In contrast, the approach in the conventional method tends to maintain a maximum distance from obstacles, resulting in a deviation to the opposite side of the road center line. The lack of constraints on the deviation from the road center line, coupled with the influence of $J_{\text{jerk}}$, results in a slightly larger turning radius. Similar to other traffic scenarios, this algorithm faces challenges in returning to the road center line, even on straight sections of the road after obstacle avoidance. Conversely, the trajectory planned by the algorithm in this paper judiciously selects a safe distance from obstacles while ensuring a seamless return to the center line of the road after obstacle avoidance. The mean value of $J_{\text{offset}}$ is 44.36%.

Figure 4.12 shows a comparison of $J_{\text{jerk}}$ in the junction scenario. In particular, there is a significant variance in the jerk values during obstacle avoidance. In this traffic scenario, the algorithm proposed in this paper maintains a

high level of driving comfort and reduces $J_{\text{jerk}}$ by 59.36% compared to the algorithm proposed in the conventional method.

### U-shaped road scenario

U-shaped roads represent a common and complex traffic scenario. In order to comprehensively evaluate the algorithm's performance, this study includes the creation of a U-shaped road with static obstacles of different dimensions at the pivotal curves of the U-shaped road.

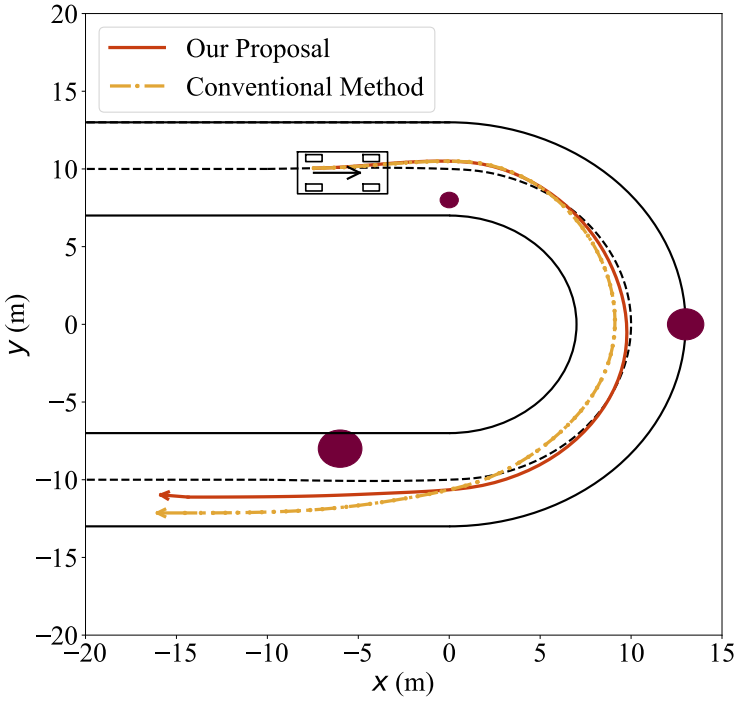



**Fig. 4.13:** Comparison of Road Center Line Offset on a U-shaped Road

Figure 4.13 illustrates the complete path taken during the continuous avoidance of static obstacles in the U-shaped road. Both algorithms show effective obstacle avoidance capabilities. However, as observed in other simulated traffic scenarios, the approach presented in the conventional method relies primarily on evaluating the safety of candidate paths based on the distance from obstacles. As a result, regardless of the size of the obstacle, the algorithm tends to maintain a safe distance by deviating as far as possible from obstacles while maintaining comfort. This results in a pronounced deviation from the center line of the road in the planned trajectory, especially at the

turning points of the U-shaped road. In contrast, this study integrates the size of obstacles into the safety evaluation of candidate paths. It chooses an appropriate deviation distance depending on the size of the obstacle and actively tries to return to the center line of the road after avoiding the obstacle.

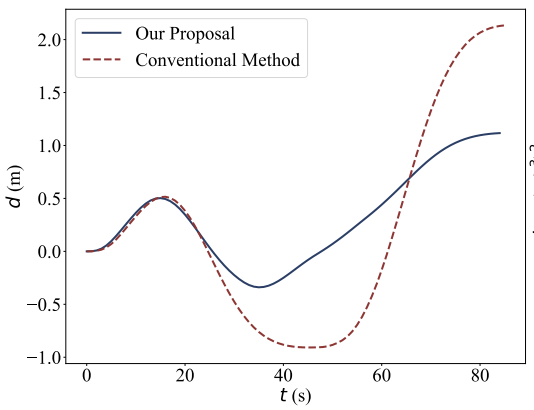



**Fig. 4.14:** Anti-deviation Comparison

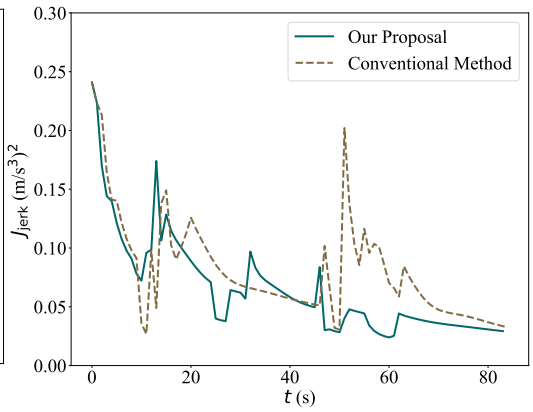



**Fig. 4.15:** Comfort Comparison

Figure 4.14 shows a comparison of the road center line deviation on the U-shaped road. It can be seen that the method proposed in this paper skilfully maintains proximity to the road center line while meeting the challenges of obstacle avoidance and turning. In contrast, the approach presented in the conventional method shows more pronounced deviations from the center line due to the lack of constraints imposed by $J_{\text{offset}}$. The mean deviation is reduced by 45.56%.

Figure 4.15 shows a comparison of the ride comfort. For the first 50 seconds, there is little difference in the jerk values between the two algorithms. However, from 50 seconds to 80 seconds, the method proposed in this paper shows a significant improvement in jerk compared to the approach advocated in the conventional method. This improvement is due to the fact that the algorithm selects a trajectory that is close to the original direction after the turn when facing the final obstacle. Conversely, the reference method always tries to maintain the maximum distance from the obstacle, resulting in increased $J_{\text{jerk}}$ due to excessive manoeuvring. The average value of $J_{\text{jerk}}$ shows a decrease of 18.60%.

## 4.8 Summary

In this chapter, the proposed local path planning algorithm for autonomous driving, based on the Frenet frame, effectively separates transverse and longitudinal motion. It generates a cluster of candidate trajectories in the $s - d$ coordinate system, considering start and end point state information. Trajectory selection is determined by a novel cost function that integrates comfort, safety and offset from the road center line, enhanced by obstacle size considerations. Comparative experiments with a reference method show that our algorithm successfully navigates various road scenarios, avoiding obstacles and returning to the center line. Remarkably, it achieves increased comfort, with reduced mean values of $J_{\text{jerk}}$ and $J_{\text{offset}}$ across different road types, showcasing its efficiency in guiding vehicles through complex environments.

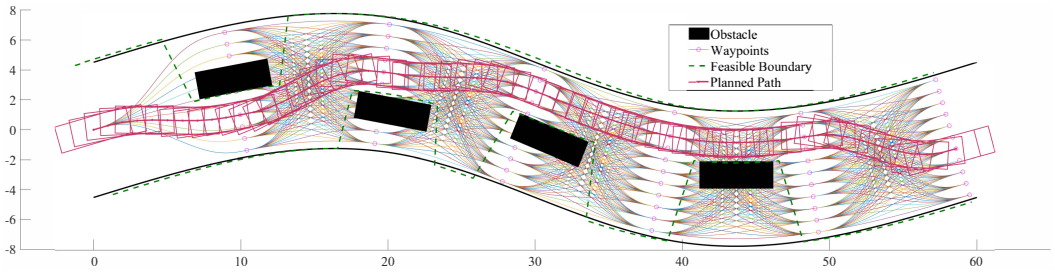
# 5

# An Adaptive Path Planning Method for Dense Obstacles via Three-Dimensional Space-aware Profiling Maps

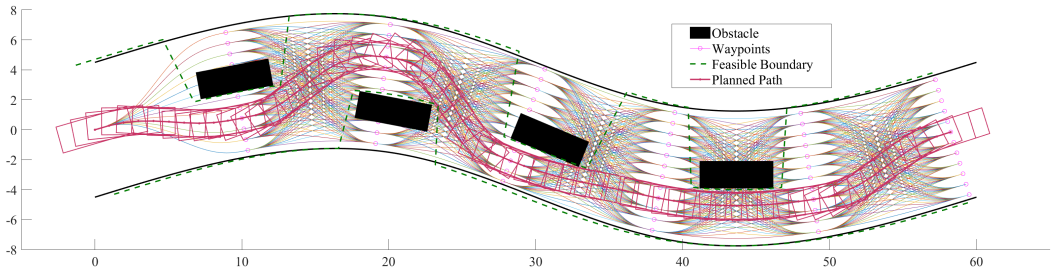
## 5.1 Background

Currently, local path planning algorithm frameworks can be categorized into three main types: lateral-longitudinal motion decoupled frameworks, STJ frameworks, and spatio-temporal decoupled frameworks, also called PVD frameworks. Among them, the PVD framework separates the complex three-dimensional trajectory planning problem into two independent parts: a two-dimensional path planning problem and a one-dimensional velocity planning problem. This innovative framework significantly reduces the computational and problem complexity associated with trajectory planning while ensuring real-time performance, making it widely adopted in practical applications. Within the framework of time-space decoupling, a common approach for structured roads is to discretize the road using uniform sampling to improve the efficiency and applicability of the planning algorithm.

The fundamental concept of the uniform sampling road method involves discretizing the path space into a set of uniformly spaced sampling points, which are subsequently interconnected through interpolation to generate a smooth trajectory. This approach serves to reduce the complexity associated with




**(a)** Sampling with 11 waypoints in $l$-direction




**(b)** Sampling with 12 waypoints in $l$-direction

**Fig. 5.1:** The conventional sampling method.

path-planning tasks and enhances computational efficiency. Nevertheless, it is imperative to recognize that this method does possess certain limitations. As illustrated in Figure 5.1, when all other parameters remain constant, modifying the number of waypoints sampled in the lateral ($l$) direction leads to varying routes for avoiding the first two obstacles. In Figure 5.1 (a), where there are 11 sampled waypoints in the lateral direction, the planned path opts to evade obstacles from the left side starting from the third obstacle. Conversely, in Figure 5.1 (b), with 12 sampled waypoints in the lateral direction, the planned path circumvents obstacles from the right side starting from the third obstacle. Therefore, this method exhibits a high sensitivity to the sampling resolution of waypoints, with even minor adjustments in the sampling resolution resulting in substantially different path-planning outcomes. In subsequent sections, we will refer to this uniform sampling road method as the conventional sampling approach.

The reason for this lies in the fact that the conventional method performs road scenario sampling without a clear purpose or subjective understanding. It relies solely on uniformly sampling road scenarios, resulting in the planned path displaying a degree of unpredictability when choosing obstacle avoidance directions. Consequently, in practice, the selection of sampling parameters heavily leans on expert experience or manual adjustment. Once

these parameters are fixed, they remain unmodifiable. This approach becomes impractical since fixed discretization parameters struggle to adapt to the constantly changing traffic conditions. Furthermore, when configuring the sampling parameters, it does not account for the distribution of obstacles on the road. This can lead to sampled waypoints being situated on or in close proximity to obstacles.

As a result, a substantial number of paths with a risk of collision may be generated without the need for collision checks. This inefficiency leads to the wastage of computational resources during subsequent collision detection steps and hinders the provision of comprehensive solution space for optimal path selection in the subsequent stages. A more sensible approach involves densely sampling regions where optimal paths are likely to exist, sparsely sampling regions near obstacles, and avoiding sampling altogether in regions occupied by obstacles.

## 5.2 Algorithm Overview

The schematic overview of our approach is depicted in Figure 5.2. Our method aims to enhance the path planning module for on-road autonomous driving, focusing specifically on spatial path planning without involving time-related velocity planning. The proposed framework adopts a hierarchical architecture with four functional layers. Initially, traffic scenarios in the Cartesian frame undergo transformation into the Frenet frame. In the Frenet frame, the $s$-axis corresponds to the direction along the road center line, and the $l$-axis represents the direction perpendicular to the road center line. This transformation conveniently converts curved road scenarios into straight road scenarios, providing an intuitive representation of the spatial relationship between the autonomous vehicle and the road. For instance, the current $s$-coordinate indicates the vehicle's accumulated distance along the road, while the $l$-coordinate conveys information about the vehicle's lateral deviation from the road center.

Following this, we model the impact of obstacles on the autonomous vehicle's trajectory, generating a 3D space-aware profiling map in the Frenet frame. Given the critical role of the sampling strategy in the $l$-direction for obstacle

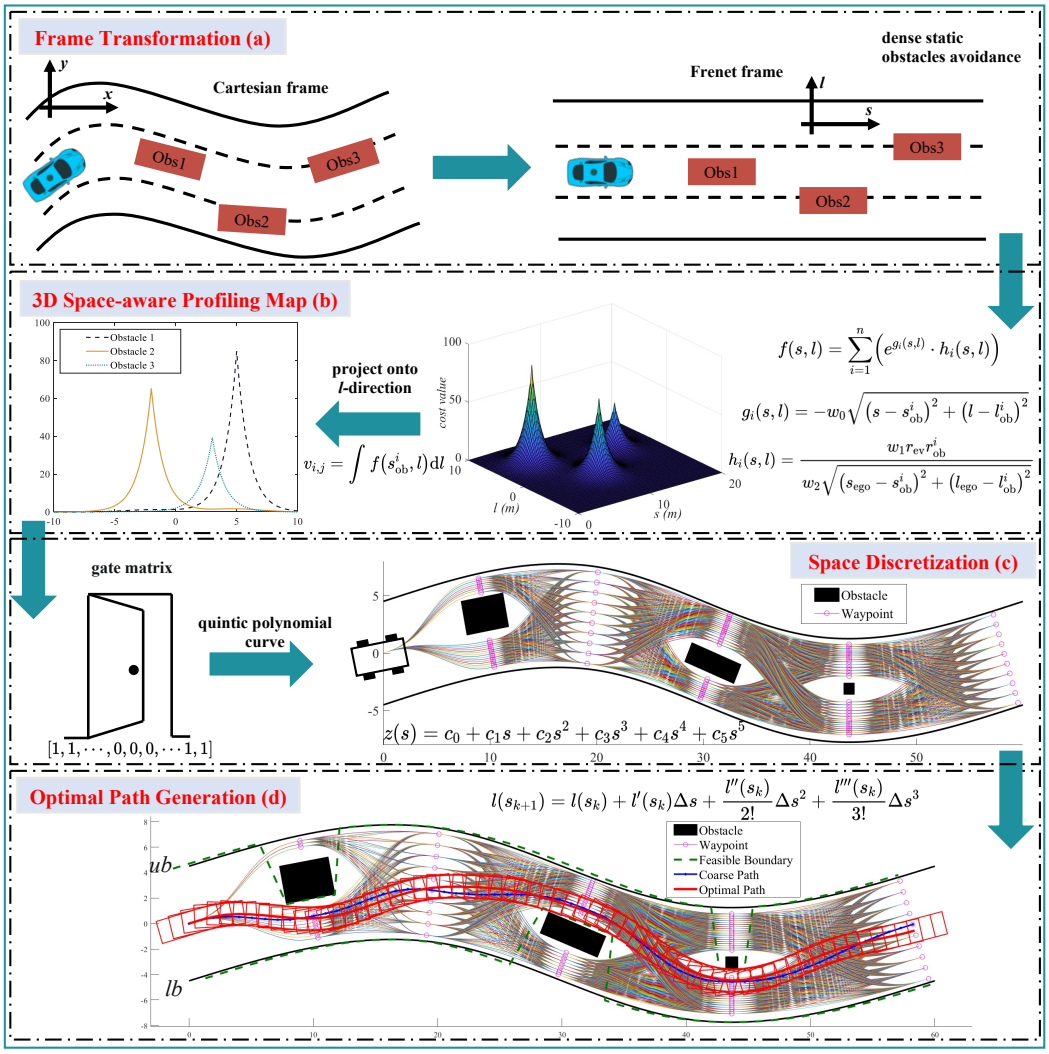



**Fig. 5.2:** The framework of the proposed adaptive path planning method.

avoidance, we project the 3D space-aware profiling map onto the $l$-axis to derive the obstacle influence curve in that direction. Additionally, through an analysis of obstacle distribution along the $l$-axis, we construct gate matrices that determine the number of samples taken in the $l$-direction by combining the obstacle influence curves. Considering the relatively minor impact of the $s$-direction on obstacle avoidance, we simplify the sampling operation in this direction by primarily relying on the distribution of obstacles in this dimension.

In the optimal path generation phase, we employ a first-coarse-to-optimal architecture. Initially, a set of collision-free paths is obtained through a collision check process. Subsequently, a customized cost function is utilized

to select the coarse solution for the optimal path. Finally, we propose a QP-based smoother to fine-tune the coarse path.

## 5.3 3D Space-aware Profiling Map

When conducting sampling within the driving environment, it is imperative to steer clear of arbitrary and aimless methodologies, as well as mechanical approaches that uniformly partition the road. Instead, a more nuanced strategy is advocated, one that takes into consideration pertinent information from the current traffic scene, including the ego vehicle's position, the distribution and dimensions of obstacles, and more. As a result, we propose an adaptive sampling methodology. The initial phase involves the modeling of the abstract driving scenario, wherein qualitative descriptions of the driving scene are translated into a quantitative cost map.

This paper addresses three pivotal factors in establishing the cost map for the existing traffic scenario. Firstly, higher cost values should be attributed to positions in closer proximity to obstacles. Secondly, obstacles nearer to the ego vehicle should wield a greater influence, contributing to elevated cost values within their respective zones. Thirdly, the dimensions of obstacles play a significant role in determining the magnitude of their impact on the corresponding regions. The modeling of the cost value for the driving scenario denoted as $f(s, l)$, can be achieved through the application of Equation 5.1:

$$f(s,l) = \sum_{i=0}^{n} \left( e^{g_i(s,l)} \cdot h_i(s,l) \right) \tag{5.1}$$

where $i$ represents the obstacle index, and $n$ denotes the total number of obstacles.

The function $g_i(s, l)$ characterizes the influence of obstacles on the driving environment, with elevated values signifying diminished driveability within the corresponding regions.

$$g_i(s,l) = -w_0 \sqrt{(s - s_{\text{ob}}^i)^2 + (l - l_{\text{ob}}^i)^2} \tag{5.2}$$

where the weight coefficient $w_0$ plays a crucial role in determining the significance of $g_i(s,l)$ within the context of $f(s,l)$. Additionally, the coordinates $(s^i_{\text{ob}}, l^i_{\text{ob}})$ represent the position of obstacle $i$ in the Frenet frame, further influencing its impact on the overall cost function.

The function $h_i(s,l)$ quantifies the impact of obstacle $i$ on the navigation of the ego vehicle:

$$h_i(s,l) = \frac{w_1 r^i_{\text{ob}} r_{\text{ev}}}{w_2 \sqrt{(s_{\text{ego}} - s^i_{\text{ob}})^2 + (l_{\text{ego}} - l^i_{\text{ob}})^2}} \tag{5.3}$$

where $r^i_{\text{ob}}$ denotes the radius of the bounding box around obstacle $i$, while $r_{\text{ev}}$ represents the radius of the bounding box encircling the ego vehicle, as illustrated in Figure 5.3. The coordinates of the ego vehicle in the Frenet frame are denoted as $(s_{\text{ego}}, l_{\text{ego}})$, and $w_1$ serves as the coefficient influencing the significance of obstacle size in the cost function. Additionally, $w_2$ is a weight coefficient impacting the consequences of the distance between obstacle $i$ and the ego vehicle.

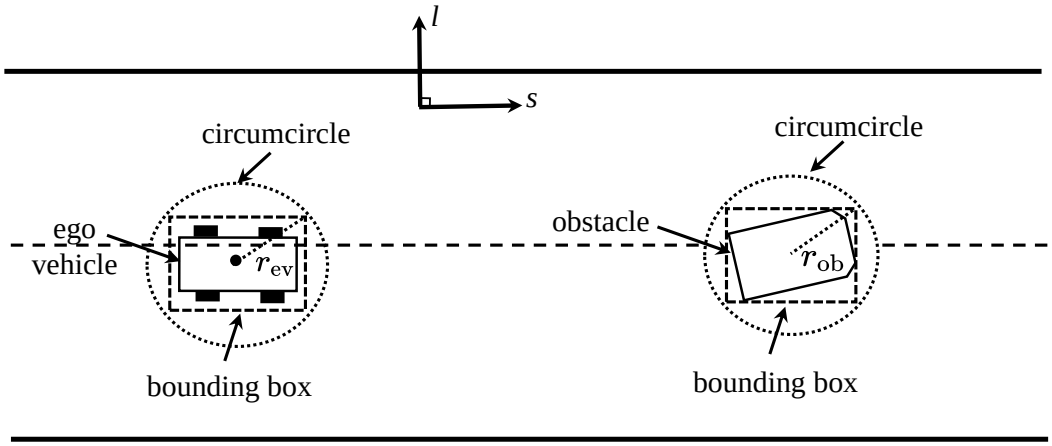



**Fig. 5.3:** An illustration of collision radius.

## 5.4 Space Discretization

In this phase, the sampling of the driving space will be conducted along both the $s$-direction and $l$-direction, guided by the 3D space-aware profiling map. Furthermore, the generation of path points in the $l$-direction will be

meticulously controlled through the implementation of the proposed gate matrix, as visually depicted in Figure 5.4.

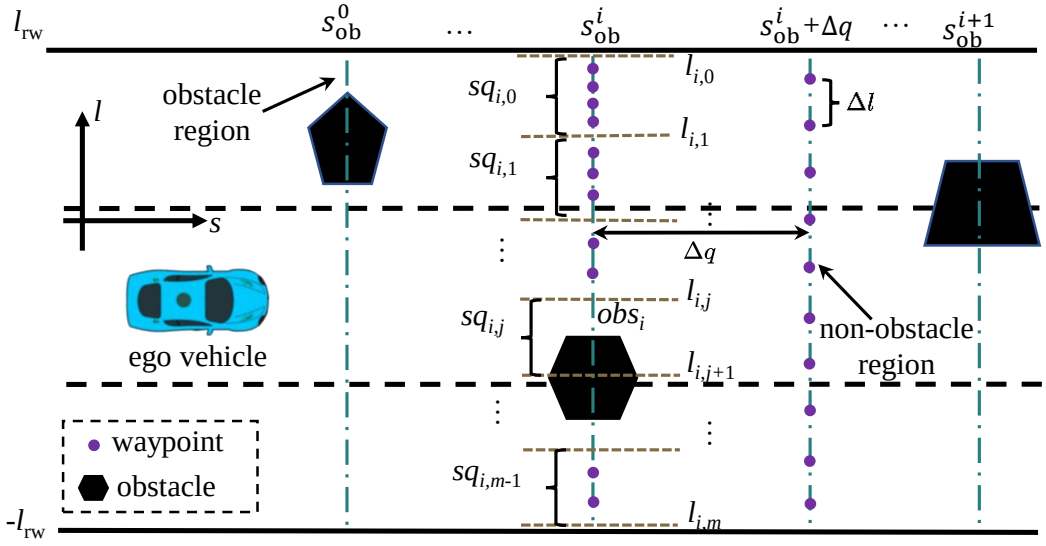



**Fig. 5.4:** The space discretization overview.

## 5.4.1 $s$-direction Sampling

Unlike the traditional approach of uniformly dividing road segments, the spatial discretization method proposed in this study takes into consideration the obstacle distribution during $s$-direction sampling. Initially, we acquire the $s$-coordinates of obstacles and perform a coarse division of the road into segments. Subsequently, we factor in the distances between adjacent obstacles in the $s$-direction. If the distance exceeds the threshold $S_r$, we employ interpolation sampling using $\Delta q$ between neighboring obstacles to ensure a rational sampling spacing and adequate sampling depth in the $s$-direction, as elucidated in Equation 5.4.

$$\begin{aligned} \mathbf{A} = &\left[s^0_{\text{ob}}, \cdots s^{i-1}_{\text{ob}}, s^i_{\text{ob}}, s^i_{\text{ob}} + \Delta q, s^i_{\text{ob}} + 2\Delta q,\right. \\ &\left. \cdots, s^{i+1}_{\text{ob}}, \cdots, s^n_{\text{ob}}\right], \\ \text{s.t.} \begin{cases} s^i_{\text{ob}} - s^{i-1}_{\text{ob}} < S_r; \\ \\ s^{i+1}_{\text{ob}} - s^i_{\text{ob}} > S_r \end{cases} \end{aligned} \tag{5.4}$$

where the matrix $\mathbf{A}$ denotes the coordinates of the sampled waypoints along the $s$-direction.

### 5.4.2 Gate Matrix

The gate matrix $\mathbf{B}$ governs the generation of waypoints in the $l$-direction, specifically targeting coordinates that intersect with obstacle positions. Initially, as depicted in Figure 5.4, the $l$-directional road segment is discretized into $m$ regions, denoted as $\text{sq}_{i,j}$. Subsequently, values are assigned to the gate matrix $\mathbf{B}$ based on the occupancy of obstacles within the corresponding regions, as denoted in Equations 5.5 and 5.6.

$$\mathbf{B} = \begin{bmatrix} b_{0,0} & b_{0,1} & \cdots & b_{0,j} & \cdots & b_{1,m-1} \\ b_{1,0} & b_{1,1} & \cdots & b_{1,j} & \cdots & b_{2,m-1} \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ b_{i,0} & b_{i,1} & \cdots & b_{i,j} & \cdots & b_{i,m-1} \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ b_{n,0} & b_{n,1} & \cdots & b_{n,j} & \cdots & b_{n,m-1} \end{bmatrix} \tag{5.5}$$

where,

$$b_{i,j} = \begin{cases} 1 & \text{obs}_i \text{ not in } \text{sq}_{i,j} \\ \\ 0 & \text{obs}_i \text{ in } \text{sq}_{i,j} \end{cases} \tag{5.6}$$

$j$ represents the index of the segmented regions.

### 5.4.3 $l$-direction Sampling

The sampling process in the $l$-direction is separated into two distinct segments: sampling in non-obstacle regions and sampling in obstacle regions. Non-obstacle regions are characterized by relatively simple traffic scenarios, resulting in a conventional uniform partitioning of the $l$-direction with $\Delta l$. Conversely, the sampling strategy in the $l$-direction, especially within

obstacle regions, plays a crucial role in determining the obstacle avoidance performance of autonomous vehicles. In this study, we project the acquired 3D space-aware profiling map in the $l$-direction and generate the cost-value curve for each obstacle as denoted in Figure 5.2 (b). Using this curve, we calculate the sum of the cost values within each region $sq_{i,j}$ to determine the number of sampling points in each region.

The rows of the matrix $\mathbf{V}$ represent the set of cost values for each region in the $l$-direction corresponding to each obstacle.

$$\mathbf{V} = \begin{bmatrix} v_{0,0} & v_{0,1} & \cdots & v_{0,j} & \cdots & v_{1,m-1} \\ v_{1,0} & v_{1,1} & \cdots & v_{1,j} & \cdots & v_{2,m-1} \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ v_{i,0} & v_{i,1} & \cdots & v_{i,j} & \cdots & v_{i,m-1} \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ v_{n,0} & v_{n,1} & \cdots & v_{n,j} & \cdots & v_{n,m-1} \end{bmatrix} \tag{5.7}$$

The calculation of the cost value $v_{i,j}$ for the region $\mathrm{sq}_{i,j}$ is illustrated as Equation 5.8:

$$v_{i,j} = \int_{l_{i,j}}^{l_{i,j+1}} f\left(s_{\mathrm{ob}}^{i}, l\right) \mathrm{d}l \tag{5.8}$$

The relationship between $l_{i,j}$ and $l_{i,j+1}$ can be expressed through Equation 5.9, establishing a connection between consecutive $l$ coordinates within the region $\mathrm{sq}_{i,j}$.

$$l_{i,j} - l_{i,j+1} = \frac{2l_{\mathrm{rw}}}{m} \tag{5.9}$$

$l_{\mathrm{rw}}$ denotes the coordinate of the upper boundary of the road in the $l$-direction.

The rows of the matrix $\mathbf{D}$ correspond to the number of waypoints assigned to each region in the $l$-direction for each obstacle.

$$\mathbf{D} = \begin{bmatrix} d_{0,0} & d_{0,1} & \cdots & d_{0,j} & \cdots & d_{1,m-1} \\ d_{1,0} & d_{1,1} & \cdots & d_{1,j} & \cdots & d_{2,m-1} \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ d_{i,0} & d_{i,1} & \cdots & d_{i,j} & \cdots & d_{i,m-1} \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ d_{n,0} & d_{n,1} & \cdots & d_{n,j} & \cdots & d_{n,m-1} \end{bmatrix} \tag{5.10}$$

The calculation of the number of waypoints allocated to each region is determined by Equation 5.11:

$$d_{i,j} = \text{Round}(\text{U} \cdot \left(1 - \frac{v_{i,j}}{\max(\mathbf{V}(i,:))}\right)) \tag{5.11}$$

where, U represents a constant denoting the fundamental sampling resolution for each region.

Ultimately, the determination of the number of sampling waypoints is regulated by the gate matrix $\mathbf{D}$ and recorded in the matrix $\mathbf{N}$:

$$\mathbf{N} = \mathbf{B} \odot \mathbf{D} \tag{5.12}$$

As depicted in Figure 5.5, the proposed method effectively conducts $l$-directional sampling for obstacles positioned at diverse locations. Importantly, within regions occupied by obstacles, no waypoints are sampled, showcasing a prudent avoidance strategy. The proximity to obstacles results in a reduction in the number of sampling points, ensuring a cautious approach in potentially challenging areas. Conversely, regions farther away from obstacles witness an increase in sampling points, reflecting a strategic emphasis on exploring open spaces. The visualized sampling outcomes in Figure 5.2 (c) reveal a deliberate distribution of waypoints around obstacles in the obstacle regions. In contrast, non-obstacle regions exhibit an even distribution along the $l$-direction. Remarkably, the majority of generated paths demonstrate collision-free trajectories with obstacles, underscoring the effectiveness of the adaptive sampling method in providing an enhanced and more efficient solution set for optimal path selection.

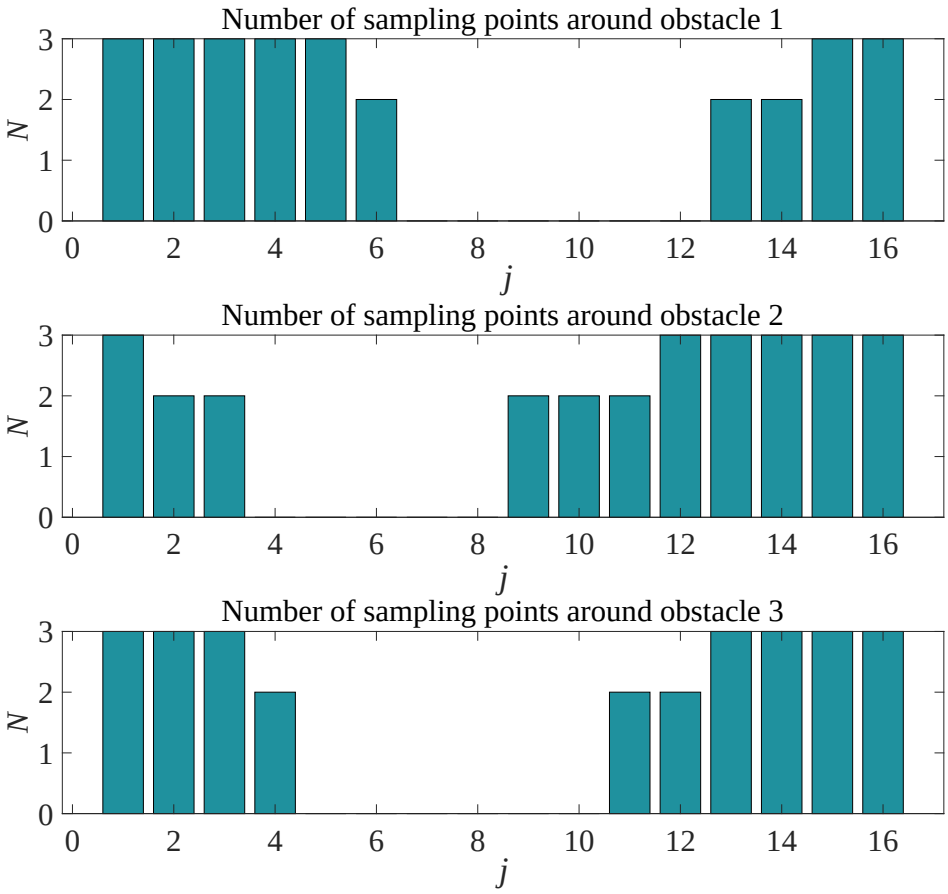



**Fig. 5.5:** The distribution of waypoints in $l$-direction around obstacles.

## 5.5 The Coarse Path Generation

In this section, we have employed a 'first-coarse-to-optimal' architecture for optimal path planning, leveraging the advantages of simplicity and assurance of an optimal solution, thereby reducing the common risks of failure associated with direct solution methods. This approach entails the initial connection of generated waypoints using curves to generate a set of potential paths. Subsequently, collision detection and assessments of vehicle kinematics are performed, eliminating paths that do not meet the specified criteria and resulting in a refined set of candidate paths. Finally, a customized cost function is utilized to select a path from the candidate path set, providing a coarse approximation of the optimal path.

### 5.5.1 Path Set Generation

Given the effective capability of the quintic polynomial curve model to accommodate complex path shapes, along with its inherent first and second-order differentiability, ensuring path continuity, we employ this model to represent the initial approximation of the planned path in the Frenet coordinate system, as denoted by Equation 5.13:

$$z(s) = c_0 + c_1 s + c_2 s^2 + c_3 s^3 + c_4 s^4 + c_5 s^5 \tag{5.13}$$

where, $c_0$, $c_1$, $c_2$, $c_3$, $c_4$, and $c_5$ are the coefficients of a quintic polynomial.

The expression $z'(s)$ denotes the first-order derivative of Equation 5.13 with respect to the parameter $s$, providing a measure of the rate of change along the path in the Frenet frame.

$$z'(s) = c_1 + 2c_2 s + 3c_3 s^2 + 4c_4 s^3 + 5c_5 s^4 \tag{5.14}$$

The term $z''(s)$ represents the second-order derivative of Equation 5.13 with respect to the parameter $s$:

$$z''(s) = 2c_2 + 6c_3 s + 12c_4 s^2 + 20c_5 s^3 \tag{5.15}$$

The calculation of the coefficient matrix involves incorporating the known conditions of the waypoints at both ends of a quintic polynomial curve into Equation 5.16, allowing for the determination of the coefficients.

$$\begin{pmatrix} c_0 \\ c_1 \\ c_2 \\ c_3 \\ c_4 \\ c_5 \end{pmatrix} = \begin{pmatrix} 1 & s_\mathrm{s} & {s_\mathrm{s}}^2 & {s_\mathrm{s}}^3 & {s_\mathrm{s}}^4 & {s_\mathrm{s}}^5 \\ 0 & 1 & 2s_\mathrm{s} & 3{s_\mathrm{s}}^2 & 4{s_\mathrm{s}}^3 & 5{s_\mathrm{s}}^4 \\ 0 & 0 & 2 & 6s_\mathrm{s} & 12{s_\mathrm{s}}^2 & 20{s_\mathrm{s}}^3 \\ 1 & s_\mathrm{e} & {s_\mathrm{e}}^2 & {s_\mathrm{e}}^3 & {s_\mathrm{e}}^4 & {s_\mathrm{e}}^5 \\ 0 & 1 & 2s_\mathrm{e} & 3{s_\mathrm{e}}^2 & 4{s_\mathrm{e}}^3 & 5{s_\mathrm{e}}^4 \\ 0 & 0 & 2 & 6s_\mathrm{e} & 12{s_\mathrm{e}}^2 & 20{s_\mathrm{e}}^3 \end{pmatrix}^{-1} \begin{pmatrix} z(s_\mathrm{s}) \\ z'(s_\mathrm{s}) \\ z''(s_\mathrm{s}) \\ z(s_\mathrm{e}) \\ z'(s_\mathrm{e}) \\ z''(s_\mathrm{e}) \end{pmatrix} \tag{5.16}$$

where, $s_s$ denotes the coordinate of the curve's starting point along the $s$-axis, while $s_e$ represents the coordinate of the curve's ending point along the $s$-axis.

### 5.5.2 Path Collision Check

We systematically perform collision detection and curvature verification to ensure the generated paths are feasible, safe, and compliant with vehicle kinematics. To enhance the accuracy of detection results, all processes are executed in the Cartesian frame.

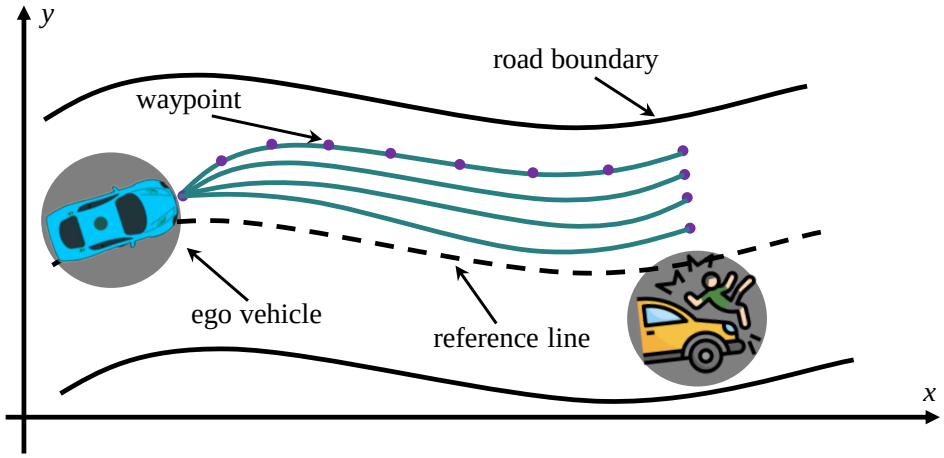



**Fig. 5.6:** The overview of path check.

The objective of the path verification process is to assess whether each waypoint along the path adheres to the specified constraints. Consequently, the collision check for the path can be articulated as follows:

$$\left(x_{h,g} - x^i_{\text{ob}}\right)^2 + \left(y_{h,g} - y^i_{\text{ob}}\right)^2 < \left(r_{\text{ev}} + r^i_{\text{ob}}\right)^2 \tag{5.17}$$

where, $(x^i_{\text{ob}}, y^i_{\text{ob}})$ represents the coordinates of obstacle $i$ in the Cartesian frame; the $(x_{h,g}, y_{h,g})$ denotes the coordinates of waypoint $h$ on path $g$ in the Cartesian frame; $r_{\text{ev}}$ signifies the radius of the circular bounding box around the ego vehicle.

By applying stringent constraints on path curvature, we ensure that the generated path adheres to the kinematic limitations of the vehicle.

$$\kappa_{h,g} \leqslant \kappa_{\max} \tag{5.18}$$

where, $\kappa_{\max}$ denotes the upper limit for curvature allowed.

### 5.5.3 Cost Function

After obtaining a set of potential navigable paths, the process of selecting the coarse solution of the optimal path introduces a subjective element. In this study, the determination of the coarse path is based on three criteria: resistance to deviation from the road center line, path smoothness, and path safety. A comprehensive cost function is developed to quantify the optimality of a given path, with a lower cost value indicating superior path performance.

$$J_g(s) = w_3 J_{offset}(s) + w_4 J_{smooth}(s) + w_5 J_{safety}(s) \tag{5.19}$$

where, $J_{\text{offset}}$, $J_{\text{smooth}}$, and $J_{\text{safety}}(s)$ denote the cost functions associated with the offset from the road center line, path smoothness, and path safety, respectively. The corresponding weight coefficients, $w_3$, $w_4$, and $w_5$, represent the emphasis placed on anti-deviation from the road center line, path smoothness, and path safety.

The anti-deviation indicator empowers the vehicle to proactively reposition itself towards the lane center following obstacle avoidance. This capability mitigates potential concerns, including inadequate space for subsequent obstacle avoidance or misjudgments involving other traffic participants.

$$J_{\text{offset}}(s) = \int_{s_{\text{init}}}^{s_{\text{end}}} z^2(s) \mathrm{d}s \tag{5.20}$$

where, $s_{\text{init}}$ represents the coordinate of the starting point of the generated path along the $s$-axis, and $s_{\text{end}}$ signifies the coordinate of the endpoint of the generated coarse path along the $s$-axis.

The criterion for path smoothness mandates that the generated path exhibits smoothness continuity and avoids minor oscillations. A path characterized by

good smoothness reduces the complexity of path tracking in the subsequent control module and ensures the comfort of autonomous driving.

$$J_{\text{smooth}}(s) = \int_{s_{\text{init}}}^{s_{\text{end}}} (z'^2(s) + z''^2(s) + z'''^2(s))\mathrm{d}s \tag{5.21}$$

where, $z'''(s)$ denotes the third-order derivative of Equation 5.13 with respect to $s$.

While all navigable paths are free from collisions, they vary in proximity to obstacles and exhibit differences in safety levels. Some paths closely approach obstacles, while others maintain a considerable distance. Moreover, paths in close proximity to larger obstacles may pose different safety considerations than those close to smaller obstacles. The assessment of path safety primarily takes into account obstacle sizes and the distance between obstacles and the planned path in the $l$-direction. The cost function for path safety is expressed as Equation 5.22.

$$J_{\text{safety}}(s) = \sum_{i=0}^{n} \frac{r_{\text{ob}}^{i}}{\int_{s_{\text{init}}}^{s_{\text{end}}} |z(s) - l_{\text{ob}}^{i}|\, ds} \tag{5.22}$$

## 5.6 The Optimal Path Generation

In the preceding section, the obtained coarse path is constructed from connected segments of quintic polynomial curves, potentially leading to discontinuities and a lack of smoothness at the connection points of adjacent curves. Furthermore, the representation of the path shape is confined by the quintic polynomial curve model. To address these challenges, this section introduces a re-modeling of the path using the Taylor series. The goal is to liberate the expression of the optimal path from the constraints imposed by the curve model. Subsequently, quadratic programming is applied to refine the coarse path, aiming to derive the optimal path.

### 5.6.1 Taylor Series Formula

The Taylor series formula, proposed by mathematicians Brook Taylor and Joseph-Louis Lagrange in the 18th century, describes the process of expanding a function around a specific point. This expansion utilizes the derivatives of the function at that point and approximates the function's values using an nth-degree polynomial. Simultaneously, it provides information about the deviation between the polynomial approximation and the actual function values.

In practical applications, the Taylor series finds wide use in the fields of mathematics and physics, particularly in calculus and approximation calculations. By truncating the Taylor series to a few initial terms, an approximate calculation of the function values can be obtained, simplifying the treatment of complex functions. However, it is essential to note that as the degree of the series increases, the accuracy of the Taylor series approximation improves, but it may introduce truncation errors. Therefore, in practical applications, the choice of an appropriate degree for the Taylor series expansion depends on the precision requirements and computational efficiency considerations. The Equation of the Taylor series formula is expressed as:

$$\begin{aligned} f(x) = f(a) + f'(a)(x-a) + \frac{f''(a)}{2!}(x-a)^2 + \\ \frac{f'''(a)}{3!}(x-a)^3 + \ldots + \frac{f^{(n)}(a)}{n!}(x-a)^n + R_n(x) \end{aligned} \tag{5.23}$$

In the Taylor series formula, $f'(a)$, $f''(a)$, $f'''(a)$, up to $f^{(n)}(a)$, represent the first, second, third, and up to the $n$th derivatives of the function evaluated at point $a$. The term $R_n(x)$ denotes the remainder, which is an infinitesimal quantity indicating the error between the $n$th-order Taylor series and the original function $f(x)$.

### 5.6.2 Path Model Formulation

Assuming the expression of the path is a function of $l$ with respect to $s$, and this function possesses third-order differentiability, if $a = s_k$, the coordinate

relationship between waypoint $(k+1)$ and waypoint $k$ can be approximated as Equation 5.6.2:

$$l(s_{k+1}) = l(s_k) + l'(s_k)\Delta s + \frac{l''(s_k)}{2!}\Delta s^2 + \frac{l'''(s_k)}{3!}\Delta s^3 \tag{5.24}$$

where $\Delta s = s_{k+1} - s_k$;

Based on the derivative definition, an approximate representation for $l'''(s_k)$ can be derived as follows:

$$l'''(s_k) = \frac{l''(s_{k+1}) - l''(s_k)}{\Delta s} \tag{5.25}$$

It is evident that the characterization of waypoints at time $k$, where $k \in [0, N]$, can be succinctly represented as $[l(s_k), l'(s_k), l''(s_k)]$. Consequently, by further manipulating Equation 5.6.2, we can articulate the interconnection between the states of waypoint $k$ and waypoint $(k+1)$ as follows:

$$\begin{pmatrix} 1 & 0 \\ \Delta s & 1 \\ \frac{1}{3}\Delta s^2 & \frac{1}{2}\Delta s \\ -1 & 0 \\ 0 & -1 \\ \frac{1}{6}\Delta s^2 & \frac{1}{2}\Delta s \end{pmatrix}^T \begin{pmatrix} l(s_k) \\ l'(s_k) \\ l''(s_k) \\ l(s_{k+1}) \\ l'(s_{k+1}) \\ l''(s_{k+1}) \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix} \tag{5.26}$$

where $T$ represents matrix transpose.

In theory, a path is composed of a series of waypoints. By adopting a chain-based approach, this paper establishes the spatial relationship between the current waypoint and the next one, thereby constructing a novel road model that is not constrained by the limitations of conventional curve models.

### 5.6.3 The Optimization Objective Function

To refine the coarse path, the objective function incorporates the results of the coarse path generation to ensure proximity to the optimal path. At the same time, the objective function also prioritizes the smoothness of the

path to avoid repeated small oscillations near the coarse solution during the process of fitting the optimal path (similar to Runge's phenomenon). In addition, similar to the cost function in Section 5.5.3, the deviation of the vehicle from the road center line is taken into account to avoid prolonged lane deviation after obstacle avoidance. The objective function is expressed as Equation 5.27.

$$J_{\text{op}}(s) = w_6 J_{\text{op}}^{\text{coarse}}(s) + w_7 J_{\text{op}}^{\text{smooth}}(s) + w_8 J_{\text{op}}^{\text{offset}}(s) \tag{5.27}$$

where, $J_{\text{op}}^{\text{coarse}}$ represents the measure of the gap between the coarse path and the optimal path, $J_{\text{op}}^{\text{offset}}$ signifies the evaluation of the deviation from the road center line, $J_{\text{op}}^{\text{smoothness}}$ indicates the assessment of the smoothness of the optimal path, and $w_6$, $w_7$, and $w_8$ are the coefficients corresponding to the three indicators, respectively.

The objective of the indicator $J_{\text{op}}^{\text{coarse}}$ is to minimize the discrepancy between the coarse path and the optimal path, as expressed in Equation 5.28.

$$J_{\text{op}}^{\text{coarse}}(s) = \sum \left(l(s_k) - z(s_k)\right)^2 \tag{5.28}$$

The objective of the indicator $J_{\text{op}}^{\text{smoothness}}$ is to ensure the smoothness and continuity of the optimal path. It achieves this by imposing constraints on the first-order, second-order, and third-order derivatives of the function $l$. The goal is to minimize the change amplitude of the planned path in the $l$-direction and to reduce the change frequency as much as possible.

$$J_{\text{op}}^{\text{smooth}}(s) = \sum \left(l'^2(s_k) + l''^2(s_k) + l'''^2(s_k)\right) \tag{5.29}$$

The objective of the indicator $J_{\text{op}}^{\text{offset}}$ is formulated as shown in Equation 5.30 to minimize the deviation from the road center line of the optimal path.

$$J_{\text{op}}^{\text{offset}}(s) = \sum l^2(s_k) \tag{5.30}$$

### 5.6.4 Path Constraint

Our approach centers on restricting the initial waypoint coordinates of the intended path and the coordinate ranges of waypoints in the $l$- direction. Assuming synchronization between the planning and control modules and perfect tracking by the control module of the previously planned path, the initial waypoint of the planned path coincides with the current position of the autonomous vehicle, as expressed in Equation 5.31.

$$\begin{cases} s_0 = s_{\text{ev}} \\ \\ l\left(s_0\right) = l_{\text{ev}} \end{cases} \tag{5.31}$$

where $(s_{\text{ev}}, l_{\text{ev}})$ are the coordinates of the ego vehicle in the Frenet frame.

As illustrated in Figure 5.2 (d), a navigable tube can be derived from the coarse path's trajectory and the spatial distribution of obstacles. Consequently, the upper boundary $ub_k$ and lower boundary $lb_k$ of this navigable tube delineate the range of coordinates for the optimal path in the $l$-direction.

$$\begin{cases} l\left(s_k\right) \leqslant ub_k - \frac{w_{\text{ev}}}{2} \\ \\ l\left(s_k\right) \geqslant lb_k + \frac{w_{\text{ev}}}{2} \end{cases} \tag{5.32}$$

where $w_{\text{ev}}$ is the width of the ego vehicle.

## 5.7 Simulation and Discussion

In the domain of path planning research, it is conventionally assumed that fundamental elements such as map data, obstacle and ego vehicle localization, and detection information encompassing obstacle dimensions, road lanes, traffic lights, and the presence of other traffic participants have been reliably obtained. This assumption abstracts away the intricate details of the implementation of front-end module functionalities within autonomous driving systems. To assess the efficacy of spatial path planning, a comparative

analysis is conducted between the traditional approaches employed by Lim et al. [104] and the adaptive methodology introduced in this paper. The simulations are executed using Matlab on an Intel Core i7 processor running at 3.00 GHz. Essential parametric configurations are delineated in Table 5.1.

**Tab. 5.1:** Basic Experimental Parameters.

| Parameter | Description | Value |
|---|---|---|
| $l_{\text{rw}}$ (m) | Road upper boundary | 5.00 |
| $r_{\text{ev}}$ (m) | Vehicle collision radius | 2.66 |
| $\Delta q$ (m) | Interpolation interval in $s$-direction | 10.00 |
| $\Delta l$ (m) | Interpolation interval in $l$-direction | 0.20 |
| $k_{\text{max}}$ (1/m) | Maximum curvature | 0.50 |
| $w_{\text{ev}}$ (m) | Width of ego vehicle | 1.80 |
| $w_0$ | Weight of obstacle position | 0.80 |
| $w_1$ | Weight of obstacle size | 1.00 |
| $w_2$ | weight of the impact of obstacles on vehicle | 0.04 |
| $w_3$ | Weight of $J_{\text{offset}}$ | 0.30 |
| $w_4$ | Weight of $J_{\text{smooth}}$ | 0.40 |
| $w_5$ | Weight of $J_{\text{safety}}$ | 0.30 |
| $w_6$ | Weight of $J_{\text{op}}^{\text{coarse}}$ | 1500 |
| $w_7$ | Weight of $J_{\text{op}}^{\text{smooth}}$ | 1000 |
| $w_8$ | Weight of $J_{\text{offset}}$ | 100 |

In the simulation phase, we initially explore the correlation between the quantity of sampled waypoints and the resulting number of generated candidate paths, taking into account the varying number of obstacles. This analysis is conducted for both the traditional methodology and our proposed approach. Furthermore, we evaluate the efficiency of the process for generating the solution set of navigable paths, making a comparative assessment between the conventional method and our proposed methodology. Finally, we juxtapose the optimal paths produced by the two distinct approaches and assess the average running time in scenarios involving consistent and disparate obstacle sizes.

### 5.7.1 Sampling Performance Comparison

As depicted in Figure 5.7, the conventional method maintains a constant number of sampling points, irrespective of variations in the number of obstacles

within the traffic scenario. In contrast, our proposed adaptive method exhibits dynamic adjustments in the number of sampling points corresponding to changes in the obstacle count. When the number of obstacles is less than three, the adaptive method samples fewer waypoints compared to traditional methods. At the same time, it increases the number of sampled waypoints in scenarios with four obstacles. These findings demonstrate that the adaptive algorithm can dynamically adapt to varying traffic scenarios by adjusting the quantity of sampled waypoints. With an increasing number of obstacles and growing complexity in road scenes, the increment in sampling points enables a more accurate evaluation of discretized traffic scenarios, actively enhancing preparedness for path planning tasks.

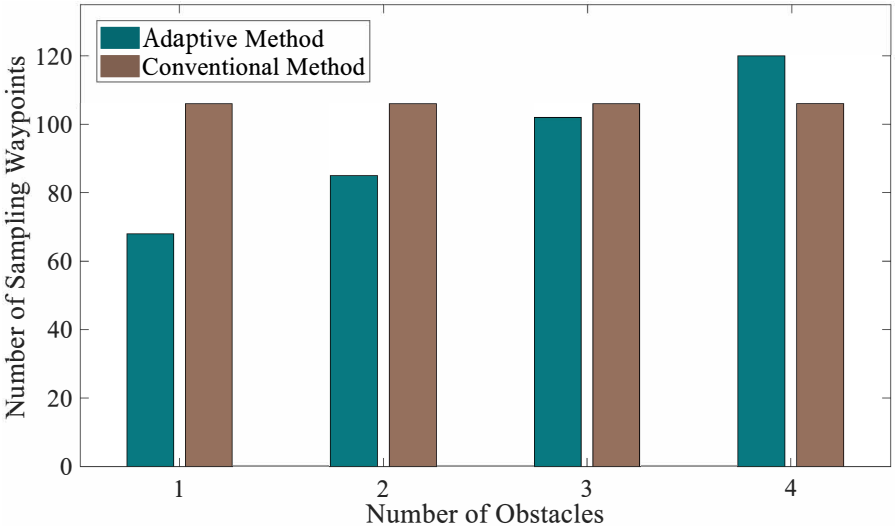



**Fig. 5.7:** The number of sampling waypoints varies with different numbers of obstacles.

Figure 5.8 provides a comparison between the conventional method and the adaptive approach proposed in this paper in terms of the number of generated paths when facing variations in obstacle quantity. In the conventional method, which does not dynamically adjust the number and distribution of sampling waypoints based on changes in road scenes, the count of candidate paths remains constant. Contrarily, as illustrated in Figure 5.7, the adaptive method generates an increasing number of candidate paths with a growing number of obstacles. Notably, even in road scenarios featuring two or three obstacles, the adaptive method produces more paths compared to conventional methods despite having fewer sampling points than traditional methods. This is attributed to the adaptive method's capability to adjust the distribution of

sampling waypoints in response to changes in road scenarios, effectively allocating limited sampling resources to areas with obstacles. Consequently, this leads to a more extensive and precise set of paths for obstacle avoidance selection.

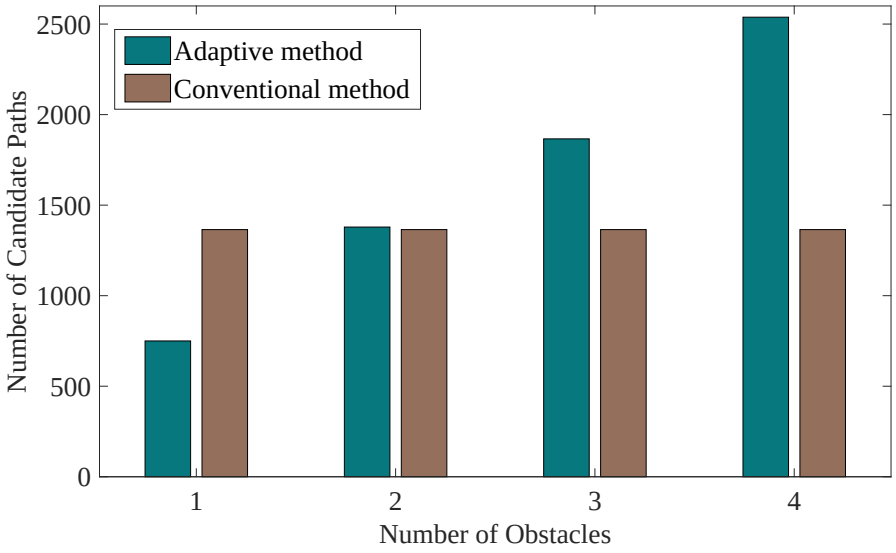



**Fig. 5.8:** The number of generated paths varies with different numbers of obstacles.

As the goal of spatial sampling is to establish a comprehensive and efficient solution space for coarse path selection, the objective is to generate paths encompassing as many navigable road segments as possible. We assess the efficacy of the sampling process using Equation 5.33.

$$E = \frac{\alpha}{\beta} \tag{5.33}$$

where $E$ is the effectiveness rate; $\alpha$ is the number of collision-free paths; $\beta$ is the number of all generated paths.

As illustrated in Table 5.2, with an increase in the number of obstacles, the conventional method exhibits a rise in the generation of collision paths and a substantial decrease in sampling effectiveness. Specifically, when faced with four obstacles, the sampling effectiveness plunges to 54.14%. In contrast, the proposed adaptive method shows a minor increase in the number of collision paths as the obstacle count rises in the road scene, maintaining a robust sampling effectiveness of 99.61% in a traffic scenario with four obstacles. This divergence stems from the fact that the conventional method, during sampling, disregards the presence of obstacles and samples the entire road.

As the number of obstacles grows, an increasing number of waypoints are sampled within or near obstacles, leading to the generation of numerous collision paths. These collision paths are devoid of meaning and utility for generating optimal paths, squandering valuable computational resources. Conversely, the adaptive method takes into account the distribution of obstacles in the road scene, avoiding inefficient waypoint sampling in areas occupied by or near obstacles. Consequently, the generated set of paths is predominantly collision-free, making optimal use of limited computational resources and furnishing a substantial solution for optimal path selection.

**Tab. 5.2:** Analysis of Spatial Sampling Effectiveness.

| | Number of Obstacles | Number of Collision Paths | Effectiveness |
|---|---|---|---|
| 1 | Conventional | 196 | 85.64% |
| | Adaptive | 0 | 100.00% |
| 2 | Conventional | 345 | 74.73% |
| | Adaptive | 2 | 99.85% |
| 3 | Conventional | 455 | 66.67% |
| | Adaptive | 6 | 99.62% |
| 4 | Conventional | 626 | 54.14% |
| | Adaptive | 10 | 99.61% |

### 5.7.2 The Optimal Path Comparison (same size of obstacles)

As depicted in Figure 5.9, the optimal paths chosen by the conventional and adaptive methods proposed in this paper differ due to different sampling approaches. In the segment between 15 m and 30 m, where the second and third obstacles are in close proximity, the conventional method, by choosing to avoid the third obstacle from its right side, requires a significant change in the current driving direction, resulting in reduced smoothness. Conversely, the adaptive method, by choosing to avoid from the left, better maintains the current driving direction, resulting in a smoother path. As the adaptive method focuses on the area where obstacles are located, it strategically samples the road based on the influence value of the obstacles. In contrast,

the conventional method neglects the positional information of the obstacles, resulting in under-sampling of the area occupied by obstacles. Consequently, this deficiency leads to insufficient smoothness in this section of the path. Moreover, in the segment between 30 m and 50 m, influenced by $J_{offset}$, the conventional approach gradually shifts the path towards the centre of the road. It initiates an avoidance action when approaching the fourth obstacle, causing a slight oscillation in the $y$ direction. In contrast, although the adaptive algorithm chooses to avoid the obstacle from the right side of the fourth obstacle, it effectively uses the obstacle avoidance space between the two obstacles, avoiding excessive turning angles and ensuring smoothness at the same time.

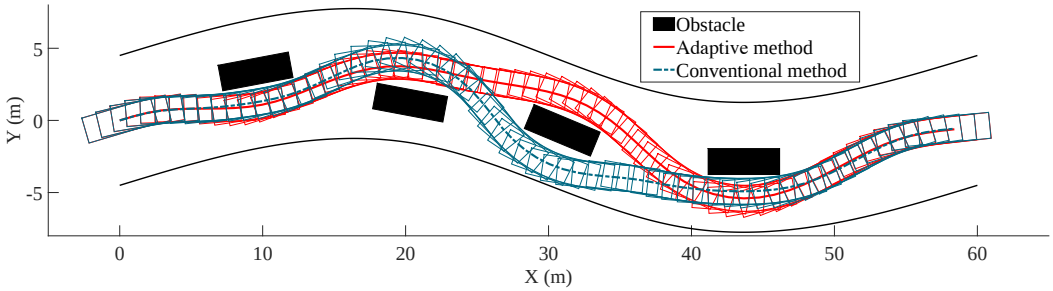



**Fig. 5.9:** Optimal path comparison (same size of obstacles).

As indicated in Table 5.3, the optimal path computed by the conventional method spans a total length of 65.75 m, whereas the adaptive method's planned path measures 62.27 m, marking a 3.48% reduction compared to the conventional method. Given that shorter paths result in faster travel times to the destination and lower energy consumption, this highlights the adaptive method's path as more economical and efficient. Additionally, the maximum heading angle for the path produced by the conventional method is 0.91 rad. In contrast, the adaptive method's path exhibits a maximum heading angle of 0.74 rad, representing an 18.68% reduction compared to the conventional method. This finding indicates that the path generated by the adaptive method is smoother, offering enhanced ride comfort. Furthermore, the average run time for the conventional method is 231.43 ms. As the adaptive method generates more candidate paths, the average run time is 238.27 ms, reflecting a slight increase of 2.96%.

**Tab. 5.3:** Performance Comparison of Length, Heading, and Run Time (same size of obstacles).

| Algorithm | Length (m) | Decline Rate (%) | Max Heading (rad) | Decline Rate (%) | Run Time (ms) | Decline Rate (%) |
|---|---|---|---|---|---|---|
| Conventional | 65.75 | 3.48% | 0.91 | 18.68% | 231.43 | -2.96% |
| Adaptive | 62.27 | | 0.74 | | 238.27 | |

### 5.7.3 The Optimal Path Comparison (different sizes of obstacles)

In this particular scenario, we configure the four obstacles with distinct sizes. Details regarding the lengths, widths of the bounding boxes, and the collision radius $r_{\text{ob}}$ for each obstacle are presented in Table 5.4.

**Tab. 5.4:** Obstacle Parameters.

| Obstacle | Length (m) | Width (m) | $r_{\text{ob}}$ (m) |
|---|---|---|---|
| $No.$ 1 | 4.00 | 3.00 | 2.50 |
| $No.$ 2 | 2.69 | 1.66 | 1.58 |
| $No.$ 3 | 5.02 | 1.96 | 2.70 |
| $No.$ 4 | 1.00 | 1.00 | 0.71 |

In Figure 5.10, the comparison of optimal paths, as influenced by variations in obstacle dimensions, reveals a consistent routing pattern for both methods, contrasting with the scenario depicted in Figure 5.9. Within the spatial range of 5 m to 15 m, corresponding to the enlargement of the initial obstacle, both the adaptive and conventional approaches demonstrate an increased steering angle favoring obstacle traversal from the right. This adjustment stems from the diminished available space on the left side, rendering it an unsafe passage. Notably, during the evasion of the third obstacle, the conventional method persists in choosing a right-sided path. The augmented size of the first obstacle contributes to an augmented negative y-axis offset during obstacle avoidance. Concurrently, the conventional method lacks an

inherent capability to dynamically adapt its sampling resolution in response to changing traffic scenarios. As a result, the chosen avoidance position remains largely static even as the second obstacle diminishes in size. Between 15 m and 25 m, the trajectory exhibits a heightened degree of bending compared to scenarios with consistent obstacle sizes. In contrast, the adaptive method opts for a leftward path around the third obstacle, actively adjusting its sampling strategy based on varying scenarios. Notably, as the size of the fourth obstacle decreases, the adaptive method dynamically refines its avoidance strategy, resulting in a smoother path within the spatial range of 25 m to 50 m.

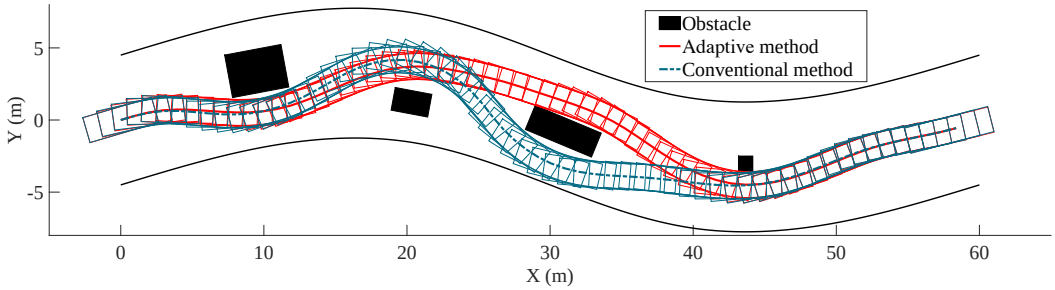



**Fig. 5.10:** Optimal path comparison (different sizes of obstacles).

As presented in Table 5.5, the total length of the optimal path generated by the conventional method exhibits a slight increase to 66.65 m. In contrast, the adaptive method yields an optimal path with a reduced total length of 61.04 m, marking a noteworthy reduction of 8.42%. This underscores the superior economy and efficiency of the adaptive method in scenarios featuring obstacles of varying sizes. Moreover, the maximum heading angle of the path generated by the conventional method reaches 0.96 rad. In comparison, the optimal path generated by the adaptive method demonstrates a substantial reduction in the maximum heading angle, registering at 0.67 rad, reflecting a significant decrease of 30.21%. By integrating this observation with the analysis of Figure 5.9, it becomes apparent that the path produced by the conventional method experiences increased curvature between 15 m and 20 m, contributing to a larger maximum heading angle. Conversely, due to the diminished size of the fourth obstacle, the path generated by the adaptive method exhibits increased smoothness between 40 m and 50 m, resulting in a reduction in the maximum heading angle. However, in this context, the average running time of the conventional method stands at 231.91 ms, whereas the adaptive method consumes 242.12 ms. This can be attributed to the reduction in obstacle size, leading to a decrease in occupied space.

Consequently, more sampling waypoints are generated in the navigable area, contributing to the generation of additional candidate paths and a subsequent increase in the running time by 4.40%.

**Tab. 5.5:** Performance Comparison of Length, Heading, and Run Time (different sizes of obstacles).

| Algorithm | Length (m) | Decline Rate (%) | Max Heading (rad) | Decline Rate (%) | Run Time (ms) | Decline Rate (%) |
|---|---|---|---|---|---|---|
| Conventional | 66.65 | 8.42% | 0.96 | 30.21% | 231.91 | -4.40% |
| Adaptive | 61.04 | | 0.67 | | 242.12 | |

## 5.8 Summary

In this chapter, we present a pioneering path planning framework that follows a coarse-to-optimal approach. Initially, we introduce an adaptive sampling method to construct a 3D space-aware profiling map in the Frenet frame, enabling the quantification of abstract traffic scenarios. This adaptive sampling method creates a cost map that guides waypoint generation by incorporating obstacle distributions in the $s$-direction and cost values in the $l$-direction. Subsequently, a comprehensive cost function is designed to derive the coarse path, taking into account factors such as ride comfort, anti-deviation from the center of the road, and path safety. Finally, a novel path model is re-established based on the Taylor series, and a quadratic programming approach is employed for fine-tuning the coarse path to achieve the optimal path.

Experimental results demonstrate the efficacy of the proposed adaptive sampling method in dynamically adjusting the sampling strategy based on changes in the number and distribution of obstacles on the road. The sampling effectiveness rate significantly improves to 100%, 99.85%, 99.68%, and 99.61% for one-obstacle, two-obstacle, three-obstacle, and four-obstacle scenarios, respectively. Comparative analysis with the conventional method

reveals a reduction in generated path length by 3.48% and 8.42% and a decrease in the maximum heading angle by 18.68% and 30.21% in scenarios involving obstacles of the same and different sizes.

# 6

# Faster Trajectory Planning for Lane Change Scenarios with Dynamic Environment

## 6.1 Traffic Scenario Introduction

Lane-changing scenarios are a common aspect of driving. In this chapter, we will focus on the lane-changing scenario shown in Figure 6.1. In contrast to avoiding static obstacles, in this scenario, autonomous vehicles must not only complete the lane-changing manoeuvre but also effectively navigate around vehicles approaching from behind to avoid collisions. This requires not only planning a feasible path in spatial dimensions but also meticulously planning the temporal motion of the autonomous vehicle. Lane-changing is therefore not just a simple task, but a comprehensive test of the trajectory planning capabilities of autonomous driving systems.

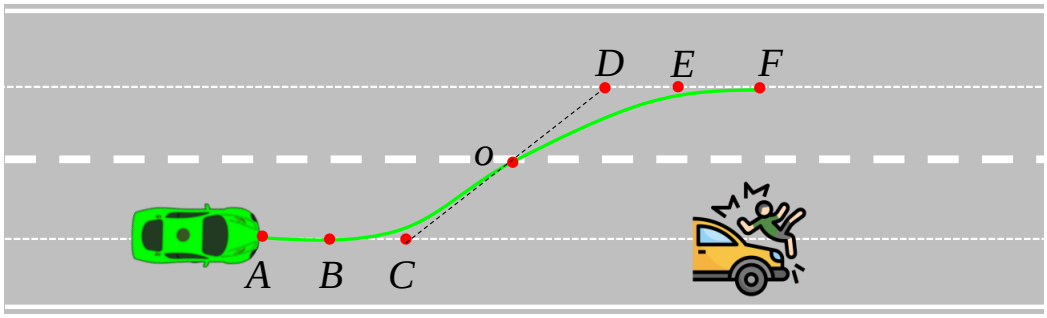



**Fig. 6.1:** An illustration of the lane change scenario with B-spline curve.

## 6.2 Background

The trajectory planning framework, known as PVD, strategically breaks down the complex three-dimensional trajectory problem into a more manageable two-dimensional path planning and a streamlined one-dimensional velocity planning [105]. This deliberate decomposition serves to significantly reduce complexity, thereby ensuring that the system is capable of real-time performance.

When changing lanes, careful selection of the curve model in path planning is crucial. This is not only because the curve model must effectively guide the vehicle through the lane change manoeuvre, but more importantly, it must ensure that the steering wheel returns smoothly to its pre-lane change position. At the same time, the feasibility of the planned path must be considered to ensure that the control module can follow it. This complex task involves not only the planning of the vehicle's path in the plane but also a comprehensive assessment of factors such as the continuity, smoothness, and operability of the vehicle's steering. However, in many current studies, this critical requirement is often overlooked in the selection of curve models. Researchers typically prioritize indicators such as path smoothness and length in path planning, neglecting key elements such as continuity, smoothness and operability in the vehicle steering process. It is therefore important to carefully select curve models that take these critical elements into account. This nuanced selection helps to improve the motion control performance of autonomous driving systems during lane changes, ensuring the smoothness and safety of lane change manoeuvres. As a result, further optimization of path planning algorithms is essential to better meet the requirements of efficient lane changing in complex driving scenarios.

Moreover, the predominant and widely adopted approach for speed planning involves the initial exploration of the $s - T$ graph using the DP algorithm, followed by the generation of a convex space. Subsequently, the QP algorithm is employed to ascertain the optimal speed profile within this convex space. Nevertheless, this method predominantly addresses the speed planning challenge through a purely mathematical lens. In the process of constructing the convex space, it overlooks the current traffic scenario, as well as the dynamic motion state and kinematic characteristics of the vehicle. This oversight

complicates a problem that could be relatively straightforward, resulting in an inefficient utilization of computational resources.

## 6.3 Path Planning

Local path planning refers to the process in a dynamic traffic environment where, utilizing onboard sensors to comprehend the surrounding road conditions, the motion planner extracts the interactive relationships between the vehicle and obstacles. After reasonably predicting their behaviors, the motion planner establishes a local path from the current position to the target position, ensuring the safe arrival of the autonomous vehicle. Path planning can be formulated as an optimization problem under a set of metrics and constraints, with the objective of finding the optimal path that meets conditions such as the shortest path length and smoothness while ensuring collision-free trajectories. Therefore, the key lies in selecting a suitable curve to describe the path.

### 6.3.1 Bessel Curve

The Bezier curve is defined by a series of control points, where the state of the curve is determined by these points. By sequentially connecting these control points, a control polygon is drawn. The Bezier curve is then enclosed by the convex hull of this polygon, which approximates the shape of the polygon itself [106].

Consider a set of control points $P_0, P_1, P_2, ..., P_n$, the Bezier curve is defined as follows:

$$C(t) = \sum_{i=0}^{n} B_i^n(t) P_i, t \in [0, 1] \tag{6.1}$$

$$B_i^n(t) = \frac{n!}{i!(n-i)!} t^i (1-t)^{n-i}, i \in \{0, 1, \ldots, n\} \tag{6.2}$$

where $t$ is the normalized time variable; $P_i$ represents the control points, and $B_i^n(t)$ denotes the base function.

The curve starts from the first control point and ends at the last control point, with the curve tangentially touching the first and last edges of the characteristic polygon at the endpoints. The shape, length, and smoothness of the curve are significantly influenced by these control points. In other words, the choice of control points determines the performance of the generated curve.

Path planning requires a minimum of $4$ control points to generate a path that meets the requirements of continuous position and direction changes, as well as smoothness. However, once the number of control points is set, the order of the path becomes fixed and a certain degree of flexibility is lost. If there are a large number of control points, especially at a higher order, the control influence on the curve is significantly reduced. In addition, the curve cannot be adjusted locally, as changing the position of a single control point will have a significant effect on the whole curve, making it less convenient and flexible.

### 6.3.2 B-spline Curve

The B-spline curve is a linear combination of B-spline basis functions. It not only retains the advantages of Bezier curves but also overcomes the disadvantage of not allowing local modifications. It also effectively addresses the issue of curvature continuity at curve junctions when describing complex shapes. It is considered to be a flexible type of curve [107].

Given a set of control points $P_0, P_1, P_2, ..., P_n$ and a node vector $U = u_0, u_1, ..., u_m$, a B-spline curve of degree $p$ and order $(p+1)$ is defined as follows:

$$C(u) = \sum_{i=0}^{n} N_{i,p}(u) P_i \tag{6.3}$$

$$N_{i,o}(u) = \begin{cases} 1, & \text{if } u_i \leq u \leq u_{i+1} \\ 0, & \text{otherwise} \end{cases}, p = 0 \tag{6.4}$$

$$N_{i,p}(u) = \frac{(u - u_i)\, N_{i,p-1}(u)}{u_{i+p} - u_i} + \frac{(u_{i+p+1} - u)\, N_{i+1,p-1}(u)}{u_{i+p} - u_{i+1}}, p \geq 1 \tag{6.5}$$

where, $N(u)$ is the base function; $u_i$ is the node.

If a node $u_i$ appears $k$ times ($k > 1$), it is referred to as a multiple node. When the nodes are evenly distributed, the node vector $U$ determines a uniform B-spline basis; otherwise, it is non-uniform. Based on the allocation of nodes in the node vector, B-spline curves are categorized into four types: uniform B-spline curves, quasi-uniform B-spline curves, piecewise Bezier curves, and non-uniform B-spline curves.

Differing from Bezier curves, the order of a B-spline curve is independent of the number of control points. Additionally, local control of the curve is achievable through basis functions, enabling the modification of any path segment without affecting adjacent segments or altering the overall shape of the curve. This provides greater flexibility.

### 6.3.3 Dubins Curve

The Dubins curve is a composite path consisting of straight lines connecting two circular arcs or three successive tangent circular arcs. Straight lines provide the shortest linear movement, while circular arcs provide the shortest turning or angular movement. The Dubins path is the shortest path between two vectors in a plane that satisfies vehicle kinematics requirements such as minimum turning radius, forward direction, initial relative position, and direction of speed [108]. It also assumes that the vehicle is moving forward only.

To determine the shortest path, we employ the following straightforward vehicle kinematics equations:

$$\dot{x}(t) = v \cos\theta \tag{6.6}$$

$$\dot{y}(t) = v \sin\theta \tag{6.7}$$

$$\dot{\theta} = k(t) \tag{6.8}$$

$$P_s\left(x_s, y_s, \theta_s\right) \overset{r(q)}{\rightarrow} P_f\left(x_f, y_f, \theta_f\right) \tag{6.9}$$

where, $(x, y)$ represents the vehicle's position at time $t$; $v$ is the velocity; $\theta$ is the heading angle; $k(t)$ denotes the curvature of the path $r(q)$ and serves as the turning rate control, with $|k(t)| \leq k_{\text{max}}$; $P_s$ and $P_f$ indicate the initial and final positions of the vehicle, respectively.

The shortest path consists of three consecutive path segments, and based on the direction of each segment, it mainly includes the following six types:

$$D = \{LSL, RSR, RSL, LSR, RLR, LRL\} \tag{6.10}$$

where $L$ represents counter-clockwise arcing, corresponding to a left turn in the driving behavior, $R$ represents clockwise arcing, indicating a right turn and $S$ represents linear motion, i.e. straight ahead.

Due to the straightforward geometric shape and high computational efficiency of Dubins curves, they have gained widespread application in the field of path planning.

### 6.3.4 Curve Model Comparison

The curve models commonly employed in path planning encompass the Dubins curve, Sine function curve, Bessel curve, B-spline curve, and others. However, the suitability of these curve models specifically for lane-changing scenarios remains a topic insufficiently explored in current research. This study aims to address this gap by conducting a comprehensive performance comparison of the aforementioned curve models in the context of lane-changing scenarios.

As shown in Figure 6.2, the Dubins curve, the sine function curve, the Bessel curve, and the B-spline curve show competence in performing lane change manoeuvres, resulting in seamlessly contoured paths with no significant difference in their paths.

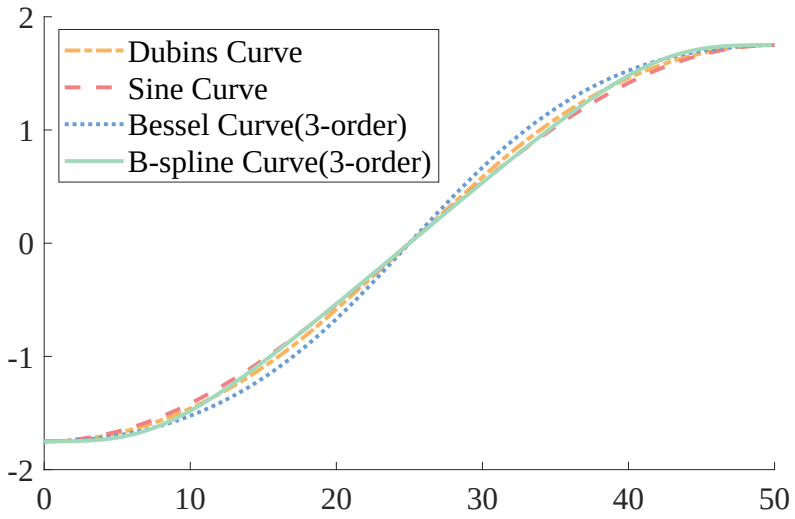



**Fig. 6.2:** Comparison of lane change paths with different curve models.

As depicted in Figure 6.3, based on the paths generated by Dubins curves, the curvature undergoes two abrupt changes. If the vehicle follows this LSR (Lane-Change Safe Route) path, driving along it during a lane change would require the vehicle to come to a stop at these discontinuities, adjust the steering, and then continue, which does not align with practical driving requirements. The lane change paths produced by sine curves, Bezier curves, and B-spline curves all exhibit continuously changing curvature. However, sine curves and Bezier paths have non-zero curvature at the target point, meaning that after completing the lane change, the steering wheel does not return to its original position, potentially causing the vehicle to deviate from the lane's centerline during subsequent travel. In contrast, B-spline curves almost avoid these two issues, making them a relatively ideal choice for path planning. Therefore, we choose the B-spline curve for lane-change path planning.

### 6.3.5 Candidate Path Set Generation

As illustrated in Figure 6.1, the lane change path comprises two segments: $\overset{\frown}{ABCO}$ and $\overset{\frown}{ODEF}$. To streamline the path-solving procedure, we assume that both $\overset{\frown}{ABCO}$ and $\overset{\frown}{ODEF}$ are centrally symmetric with respect to point $O$. Consequently, our focus narrows to determining the shape of $\overset{\frown}{ABCO}$. Given that the position of point $A$ relies on the current location of the

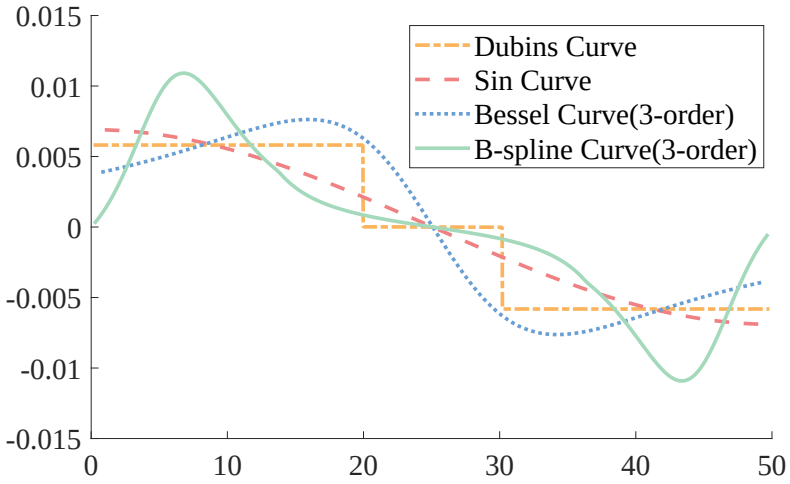



**Fig. 6.3:** Curvature comparison of different curve models in the lane change scenario.

ego vehicle, and point $B$ is the midpoint between points $A$ and $C$, the configuration of $\widehat{ABCO}$ is solely contingent on the positions of points $C$ and $O$. By manipulating the positions of points $C$ and $O$, a multitude of candidate path sets can be generated. Since there is no static obstacle in the lane-change scenario set in this chapter, therefore, we can omit the step of collision check. For dynamic obstacles, we can adjust the speed of the ego vehicle to avoid them.

### 6.3.6 Cost Function

When selecting the optimal path, we focus on the smoothness and efficiency of the path to ensure that the planned path performs well in real-world navigation or planning applications. Smoothness considerations mean that we look for paths that have relatively low curvature so that there are fewer sharp turns or bends in the path, making the experience of moving along the path more comfortable and avoiding potential mechanical or kinematic system stresses.

At the same time, we emphasize the efficiency of paths, focusing mainly on their length. In path planning, short paths often significantly increase the efficiency of navigation or transport and reduce resource consumption at the same speed situation. Therefore, we strive to minimize the total length of

paths, while maintaining a smooth selection of optimal paths. This not only improves the overall performance of the system but also reduces the time and cost of the navigation or planning process.

Ultimately, the goal is to find a path that has the best balance of curvature and length to meet the needs of real-world applications. This combination of smoothness and efficiency helps to ensure that the generated paths can be used effectively and feasibly in a variety of application scenarios. The cost function is denoted as Equation 6.11:

$$\begin{aligned} J_{\text{path}} =& w_1 \sum_{i=1}^{n-2} \left| \frac{\arctan\left(\frac{y_{i+2}-y_{i+1}}{x_{i+1}-x_{i+1}}\right) - \arctan\left(\frac{y_{i+1}-y_{i+1}}{x_{i+1}-x_i}\right)}{\sqrt{(x_{i+1}-x_i)^2 + (y_{i+1}-y_i)^2}} \right| \\ &+ w_2 \sum_{i=1}^{n-1} \sqrt{(x_{i+1}-x_i)^2 + (y_{i+1}-y_i)^2} \end{aligned} \tag{6.11}$$

$(x_i, y_i)$ denotes the coordinates of waypoint $i$ within the generated lane path; $n$ signifies the count of waypoints along the generated lane path; $w_1$ and $w_2$ stand for the weighting coefficients governing path smoothness and length, respectively.

## 6.4 Speed Planning

The trajectory planning module plays a critical role in ensuring the safety of autonomous vehicles. Traditional planning methods, which do not directly consider vehicle interaction, eliminate spatial paths that would lead to collisions with dynamic or static obstacles during collision detection. In densely populated urban road environments, this approach can lead to mutual influence between path planning and speed planning, resulting in an unstable overall planning scheme. This instability makes it difficult to ensure safe interaction with dynamic obstacles in driving scenarios. In real-world urban dynamic driving scenarios, experienced human drivers can adjust their speed while maintaining the spatial path, reducing the safety risks associated with frequent planning adjustments. In this section, we will introduce the speed planning method.

### 6.4.1 $s-T$ Graph Introduction

The $s-T$ graph is the basis for speed planning. It reflects in the time domain the influence of dynamic obstacles on the currently planned path. The construction of the $s-T$ graph is illustrated in Figure 6.4. As the ego vehicle travels along the planned path, the four corner points $(a, b, c, d)$ of an obstacle vehicle at time $t_n$ are evaluated with their respective shortest distances to the planned path $(aa', bb', cc', dd')$. If all of these distances $(aa', bb', cc', dd')$ are greater than a threshold $\omega$, this indicates that the obstacle vehicle does not currently affect the trajectory of the ego vehicle and does not pose a collision risk. Conversely, if all of these distances are less than or equal to the threshold $\omega$, it indicates that the obstacle vehicle is interfering with the ego vehicle's trajectory. In such cases, the obstacle vehicle must be projected onto the $s-T$ graph.

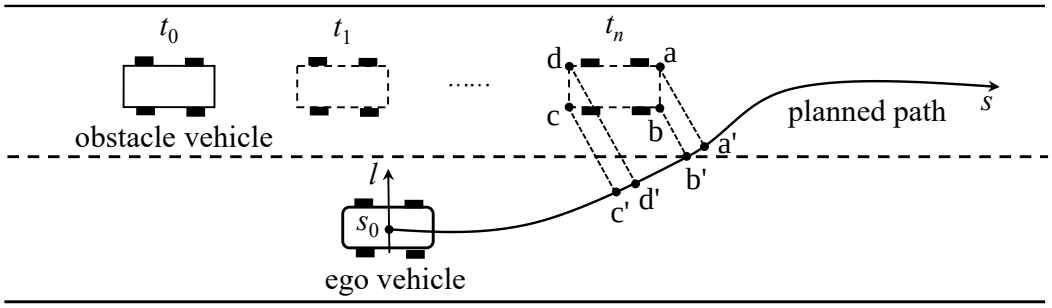



**Fig. 6.4:** Lane Change Scenario.

Figure 6.5 illustrates the $s-T$ graph, where $s_{\text{max}}$ represents the total length of the current planned path, and $t_{\text{max}}$ denotes the allowed maximum time for the current speed planning. In the region enclosed by $s_{\text{max}}$ and $t_{\text{max}}$, except for the area occupied by obstacles, the remaining space constitutes the collision-free zone, indicating the feasible solution space for speed planning. In speed planning, within the allowed planning time $t_{\text{max}}$, autonomous vehicles are not required to complete the entire planned path. Unfinished segments can be re-planned in the subsequent speed planning for a seamless trajectory.

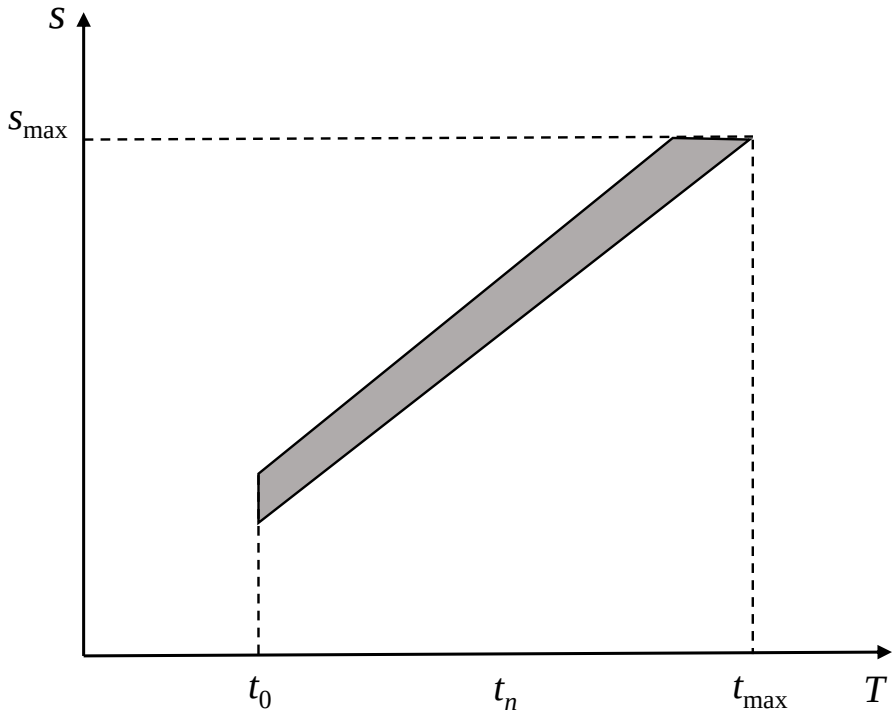



**Fig. 6.5:** Illustration of $s - T$ Graph.

### 6.4.2 Convex Space Generation Analysis

To enhance the responsiveness and rationality of the path planning process, we introduce an innovative convex space generation method. This method carefully considers both the kinematic characteristics of the vehicle and the current traffic scenario. Illustrated in Figure 6.6, two distinct convex spaces are identified: region A, situated above the obstacle area, and region B, located below the obstacle area. Region A signifies the ego vehicle merging into the lane before the rear obstacle vehicle arrives, while region B implies that the ego vehicle yields to the rear obstacle vehicle, waits for it to pass, and subsequently merges into the lane. However, due to constraints such as the ego vehicle's speed during lane change, the maximum allowable acceleration and deceleration in the vehicle's kinematics, the speed of the obstacle vehicle, and the distance from the obstacle vehicle, it often becomes challenging for regions A and B to simultaneously satisfy these constraints. Therefore, a thorough evaluation of the current traffic situation becomes imperative.

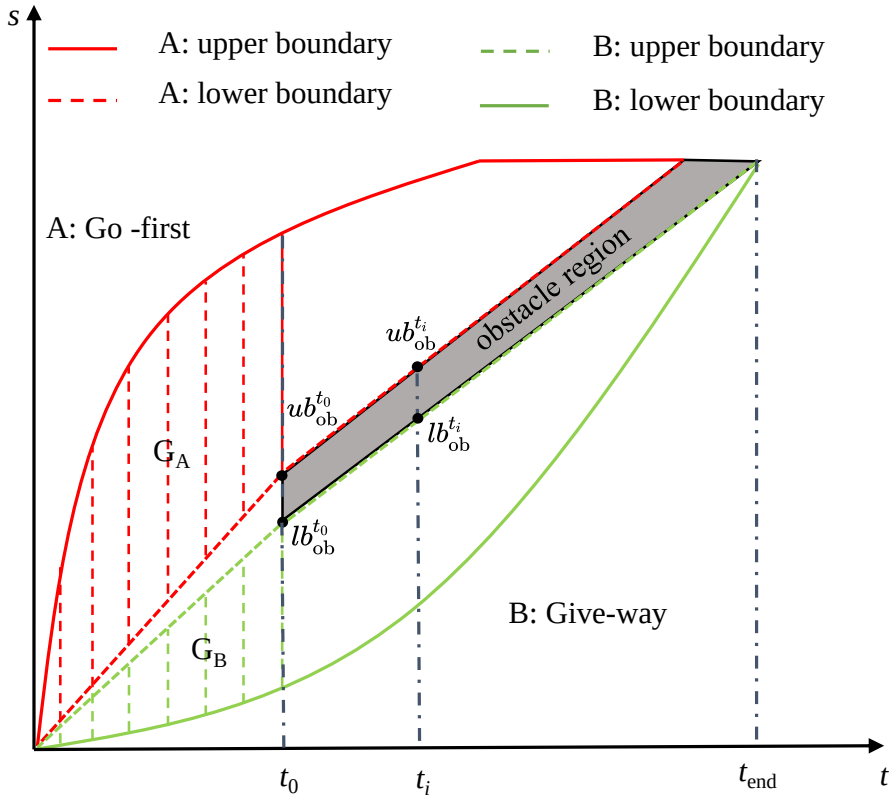



**Fig. 6.6:** An illustration of constraints and convex space on the $s$-$T$ Graph.

Under the assumption that the ego vehicle undergoes uniform acceleration or deceleration and avoids reversing, it is required to adhere to Equation 6.12 for successful lane changing before the rear obstacle vehicle arrives.

$$v_0 t_0 + \frac{1}{2}\bar{a}{t_0}^2 > ub_{ob}^{t_0} \tag{6.12}$$

where $v_0$ represents the initial speed of the ego vehicle; $t_0$ denotes the start time of the obstacle region; $\bar{a}$ is the maximum acceleration capability of the ego vehicle, and $ub_{ob}^{t_0}$ signifies the upper boundary of the obstacle region at time $t_0$.

Considering the limitations imposed by the vehicle's dynamic performance, not all regions within area A satisfy the requirements of the vehicle's dynamic capabilities due to the existence of a maximum acceleration threshold. Therefore, a meticulous determination of the upper boundary of the convex space is imperative. This determination takes into account the current speed of the vehicle and its potential to achieve maximum acceleration. Such a

thorough consideration ensures that every subregion within the convex space complies with the dynamic requirements. The upper boundary $S_{\mathrm{A}}^{\mathrm{ub}}(t_i)$ of the convex space in region A is determined by Equation 6.13, which holds true if Equation 6.12 is satisfied.

$$S_{\mathrm{A}}^{\mathrm{ub}}(t_i) = \begin{cases} v_0 t_i + \frac{1}{2}\bar{a}{t_i}^2, & v_0 t_i + \frac{1}{2}\bar{a}{t_i}^2 < \mathrm{S_L} \\ \mathrm{S_L}, & v_0 t_i + \frac{1}{2}\bar{a}{t_i}^2 > \mathrm{S_L} \end{cases} \tag{6.13}$$

where $\mathrm{S_L}$ is the length of the planned path.

The expression for the lower boundary $S_{\mathrm{A}}^{\mathrm{lb}}(t)$ of the convex space in region A is given by Equation 6.14.

$$S_{\mathrm{A}}^{\mathrm{lb}}(t_i) = \begin{cases} v_0 t_i + (\frac{ub_{\mathrm{ob}}^{t_0} - v_0 t_0}{t_0^2}){t_i}^2 & t_i < t_0 \\ ub_{\mathrm{ob}}^{t_i} & t_i > t_0 \end{cases} \tag{6.14}$$

where $ub_{\mathrm{ob}}^{t_i}$ is the upper bound of obstacle region at time $t_i$.

If the ego vehicle chooses to yield to the obstacle vehicle, it must speed down at the maximum deceleration under the current speed to allow the obstacle vehicle to pass before time $t_0$, denoted as Equation 6.15. If at time $t_0$, the ego vehicle's position along the $s$-coordinate is still greater than the obstacle's position along the $s$-coordinate, the yielding plan cannot be fulfilled, and the ego vehicle must change lanes before the obstacle vehicle arrives.

$$v_0 t_0 + \frac{1}{2}\underline{a}{t_0}^2 < lb_{\mathrm{ob}}^{t_0} \tag{6.15}$$

where $\underline{a}$ is the maximum deceleration of the ego vehicle.

The upper boundary $S_{\mathrm{B}}^{\mathrm{ub}}(t_i)$ of the convex space in region B can be represented by Equation 6.16, provided that Equation 6.15 holds.

$$S_{\mathrm{B}}^{\mathrm{ub}}(t_i) = \begin{cases} v_0 t_i + (\frac{lb_{\mathrm{ob}}^{t_0} - v_0 t_0}{t_0^2}){t_i}^2 & t_i < t_0 \\ lb_{\mathrm{ob}}^{t_i} & t_i > t_0 \end{cases} \tag{6.16}$$

Given the constraints of the vehicle dynamics and considering the maximum deceleration of the vehicle, not all areas in region B meet the requirements of vehicle deceleration. Based on the current speed and maximum deceleration,

we can determine the lower boundary of the convex space. In determining the lower boundary, we assume that the vehicle cannot perform reverse manoeuvres during the lane change process. However, based on human driving experience, we also consider the scenario where the vehicle is allowed to stop and wait for the leading vehicle to proceed during the lane change. The lower boundary $S_{\mathrm{B}}^{\mathrm{lb}}(t_i)$ of the convex space in region B can be formulated using Equation 6.17.

$$S_B^{lb}(t_i) = \begin{cases} v_0 t_i + \frac{1}{2}\underline{a}{t_i}^2 & v_0 + \underline{a}t_i > 0 \\ -\frac{1}{2}\frac{v_0^2}{\underline{a}} & v_0 + \underline{a}t_i \leq 0 \end{cases} \tag{6.17}$$

If both Equation 6.12 and 6.15 hold, it is crucial to assess the quality of the convex spaces in regions A and B. In actual lane-changing scenarios, humans typically assess the likelihood and risks of yielding or proceeding before making a lane change. If the leading obstacle is close and moving quickly, it is preferable to yield. However, if the obstacle is distant and moving slowly, attempting to proceed is considered. Therefore, the time before the arrival of the obstacle is paramount. In this paper, we employ the area enclosed by the upper and lower boundaries of the convex spaces in regions A and B within the time interval $0$ to $t_0$ as a metric, as illustrated in Figure 6.6. This metric reflects the system's ability to handle unexpected scenarios and the safety of speed planning solutions. A larger area implies more speed planning options for the ego vehicle, enhancing its capability to handle sudden situations and ensuring a safer speed planning solution. Additionally, a larger area is advantageous for the subsequent quadratic programming, preventing potential solution failures under certain conditions due to the presence of constraints.

$$\begin{cases} G_{\mathrm{A}} = \int_0^{t_0} \left( S_{\mathrm{A}}^{\mathrm{ub}}(t) - S_{\mathrm{A}}^{\mathrm{lb}}(t) \right) dt \\ G_{\mathrm{B}} = \int_0^{t_0} \left( S_{\mathrm{B}}^{\mathrm{ub}}(t) - S_{\mathrm{B}}^{\mathrm{lb}}(t) \right) dt \end{cases} \tag{6.18}$$

### 6.4.3 Speed Profile Formulation

The speed profile is modeled considering the piecewise-jerk method [109]. This method leverages the Taylor series to establish the connection between consecutive points $(t_i, s_i)$ and $(t_{i+1}, s_{i+1})$ on the velocity profile, as represented in Equation 6.19. Additionally, the piecewise-jerk method proves effective in constraining each point on the velocity profile, offering more adaptability compared to the direct employment of curve models for modeling.

$$\begin{pmatrix} 1 & 0 \\ \Delta t & 1 \\ \frac{1}{3}\Delta t^2 & \frac{1}{2}\Delta t \\ -1 & 0 \\ 0 & -1 \\ \frac{1}{6}\Delta t^2 & \frac{1}{2}\Delta t \end{pmatrix}^T \begin{pmatrix} s\left(t_i\right) \\ s'\left(t_i\right) \\ s''\left(t_i\right) \\ s\left(t_{i+1}\right) \\ s'\left(t_{i+1}\right) \\ s''\left(t_{i+1}\right) \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix} \tag{6.19}$$

where $\Delta t = t_{i+1} - t_i$; $s'(t)$, $s''(t)$, and $s'''(t)$ represent the first-order, second-order, and third-order derivatives of the speed profile function $s(t)$ concerning the time parameter $t$.

### 6.4.4 Optimization Objective Function

In the search for an optimal speed profile, we use the QP algorithm, a powerful optimization tool. The objective function, encapsulated by the Equation 6.20, plays a key role in guiding this optimization process. It meticulously incorporates crucial factors such as distance, velocity, acceleration, and jerk at each velocity point. These factors are critical in ensuring not only the smoothness of the speed profile but also compliance with vehicle dynamics and safety constraints. By carefully considering these elements, the algorithm aims to strike a balance and produce a speed profile that not only minimizes the smoothness of the speed profile but also prioritizes the safety and comfort of the vehicle and its occupants.

$$
\begin{aligned}
\min J_{\mathrm{QP}} &= w_3 \Sigma s^2\left(t_i\right)+w_4 \Sigma\left(s^{\prime}\left(t_i\right)\right)^2 \\
&\quad+w_5 \Sigma\left(s^{\prime \prime}\left(t_i\right)\right)^2+w_6 \Sigma\left(s^{\prime \prime \prime}\left(t_i\right)\right)^2 \\
\text{s.t.} & \begin{cases}
s\left(t_0\right)=0 \\
s^{\prime}\left(t_0\right)=v_0 \\
S^{\mathrm{lb}}\left(t_i\right) \leqslant s\left(t_i\right) \leq S^{\mathrm{ub}}\left(t_i\right) \\
\underline{v} \leqslant s^{\prime}\left(t_i\right) \leq \bar{v} \\
\underline{a} \leqslant s^{\prime \prime}\left(t_i\right) \leqslant \bar{a} \\
\underline{jerk} \leqslant s^{\prime \prime \prime}\left(t_i\right) \leqslant \overline{jerk}
\end{cases}
\end{aligned}
\tag{6.20}
$$

where $\bar{v}$ and $\underline{v}$ represent the upper and lower bounds of the speed, defining the maximum and minimum allowable speeds, respectively. Additionally, $\overline{jerk}$ and $\underline{jerk}$ denote the upper and lower limits of jerk, representing the maximum and minimum permissible jerk values. By incorporating these constraints into the optimization process, we ensure that the resultant speed profile adheres to the specified limits, guaranteeing a realistic and feasible representation of the vehicle's motion.

## 6.5 Experiment and Discussion

In this dedicated section, an in-depth analysis of the outcomes stemming from the proposed path planning and speed planning algorithms is presented. The simulations were meticulously conducted utilizing Matlab, leveraging the computational power of an Intel Core i9 processor operating at 3.00 GHz. Through comprehensive examination and scrutiny, the effectiveness and performance metrics of the algorithms are thoroughly evaluated to ascertain their capabilities in diverse scenarios and under varying conditions.

### 6.5.1 Path Planning Results

As illustrated in Figure 6.7, a multitude of potential paths are generated, forming a cluster of candidate trajectories. Leveraging the optimization framework defined in Equation 6.11, the algorithm identifies the optimal

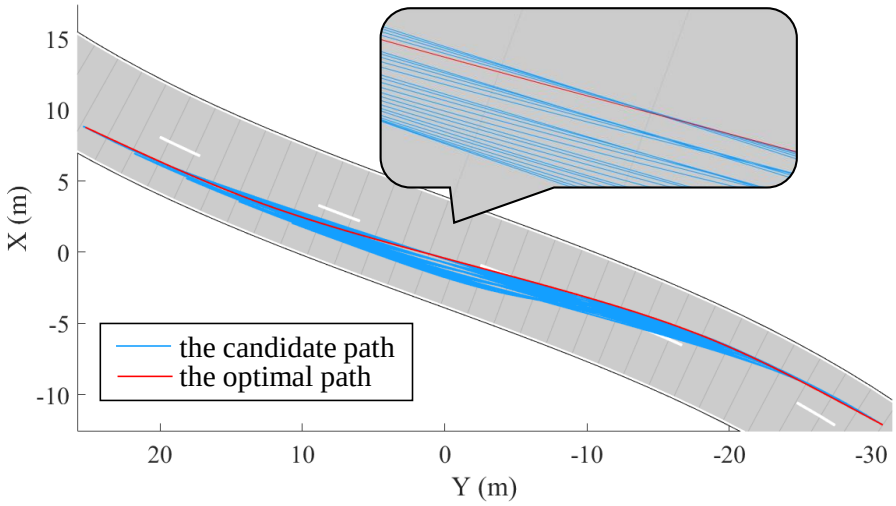



**Fig. 6.7:** The optimal path selection.

trajectory from this set of candidates. This optimal trajectory empowers the ego vehicle to execute a seamless lane change.

### 6.5.2 Speed Planning Results

In scenario $1$, with an initial ego vehicle speed of $9$ m/s, the proposed speed planning algorithm intelligently selects the region below the obstacle area as the convex space using the $s-T$ graph. This strategic decision results in the successful generation of a nuanced speed profile. As shown in Figure 6.8 (b), (c), and (d), the ego vehicle initiates a deceleration phase, yielding to the approaching obstacle vehicle. It then accelerates to execute the lane change after the obstacle vehicle has passed, culminating in a steady speed. The resulting speed, acceleration, and jerk profiles are smooth and continuous, ensuring the stability and feasibility of the lane change process.

In scenario $2$, as shown in Figure 6.9 (a), when the initial speed of the ego vehicle is $15$ m/s, the proposed speed planning algorithm strategically selects the region above the obstacle area as the convex space for exploring the optimal speed profile. Furthermore, as detailed in Figure 6.9 (b), (c), and (d), to enhance the safety and comfort of the lane change process, the ego vehicle undergoes a gradual deceleration before smoothly accelerating to merge into the target lane. Throughout this process, the speed, acceleration,

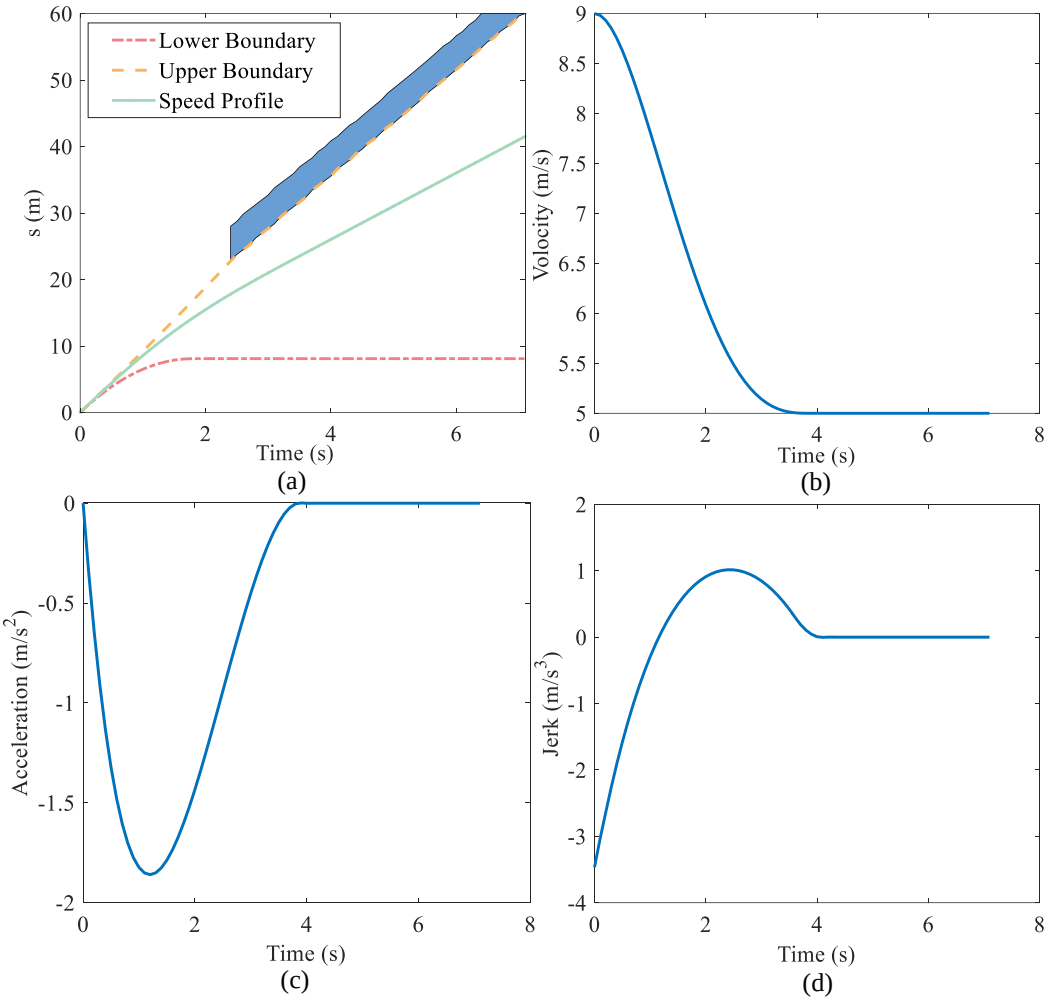



**Fig. 6.8:** The speed planning results in case 1: (a) the speed profile in $s-T$ graph; (b) the optimal velocity; (c) the optimal acceleration; (d) the optimal jerk.

and jerk profiles maintain smooth and continuous characteristics to ensure feasibility during actual driving.

As shown in Table 6.1, the conventional method (DP&QP) requires $107.4$ ms to generate the optimal speed profile. In stark contrast, our proposed method achieves the same objective in a significantly reduced time of only $72.3$ ms. This highlights a remarkable 32.7% reduction in the average running time, underlining the efficiency of our approach.

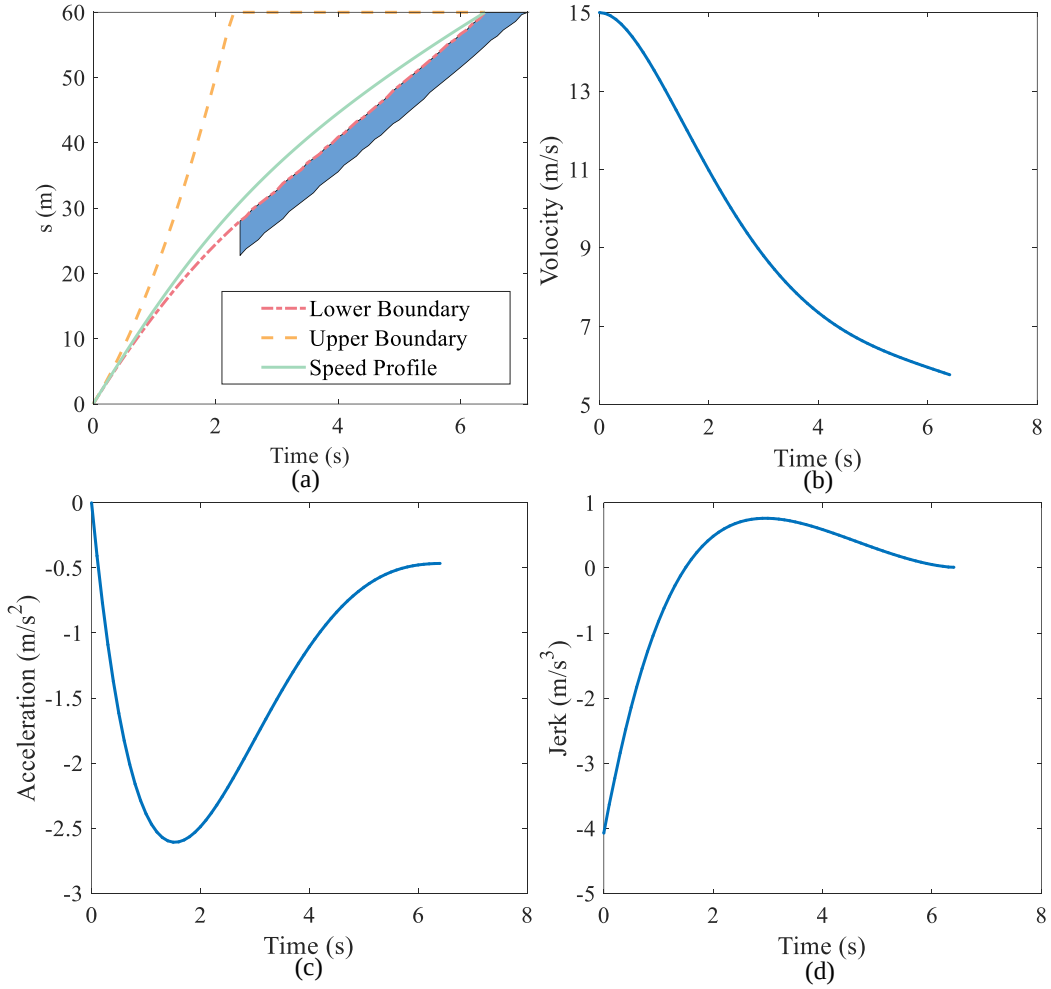



**Fig. 6.9:** The speed planning results in case 2: (a) the speed profile in $s-T$ graph; (b) the optimal velocity; (c) the optimal acceleration; (d) the optimal jerk.

## 6.6 Summary

Current research on the performance of different curve models in lane-changing scenarios is still insufficient. In addition, conventional speed planning methods (DP&QP) often ignore vehicle kinematics and traffic scenarios when generating convex spaces, leading to inefficiencies and longer runtimes. To address these issues, this chapter comprehensively evaluates the performance of Dubins, Sine, Bézier, and B-spline curves in lane change scenarios. The results show that the B-spline curve is selected as the optimal lane change path model, which ensures that the vehicle returns to its original position after lane change to prevent deviation from the road centreline.

**Tab. 6.1:** Average run-time comparison.

| Method | Time (ms) | Decline Rate |
|---|---|---|
| DP&QP | 107.4 | / |
| Ours | 72.3 | 32.7% |

In terms of speed planning, a novel method is introduced to directly generate convex spaces, taking into account the kinematics of the ego vehicle and the current traffic conditions. This innovative approach allows adaptive adjustments to the speed planning scheme based on real-time traffic conditions. Comparative analysis with traditional trajectory planning methods shows a significant runtime reduction of 32.7%, demonstrating the effectiveness of the proposed approach.

# 7

# Conclusion and Future Work

## 7.1 Conclusion

Trajectory planning is a complex optimization issue. The ultimate goal is to find a trajectory that balances factors such as safety, comfort, efficiency, and compliance with vehicle dynamics and kinematics. A successful trajectory planning algorithm requires adaptability in various traffic scenarios, enabling intelligent path planning based on the current traffic environment. However, due to the diversity of real-world traffic scenarios, expecting a single algorithm to perform excellently in all situations was challenging. Therefore, in this thesis, we proposed three planning algorithms tailored to adaptive cruising scenarios on urban roads, obstacle avoidance scenarios with dense static obstacles, and dynamic lane-changing scenarios.

For adaptive cruising scenarios on urban roads, in Chapter 4, we presented a local trajectory planning algorithm for autonomous driving that is based on the Frenet frame. The algorithm begins by decoupling transverse and longitudinal motion using the Frenet frame and generates a cluster of candidate trajectories in the $s - l$ coordinate system based on the initial and final state information. Subsequently, the algorithm identifies the optimal trajectory using a novel cost function that incorporates considerations for comfort, safety, and offset from the road center line. During the candidate trajectory selection phase, we introduce a fresh cost function for optimal trajectory selection. This cost function is meticulously crafted to comprehensively account for comfort, safety, and road center line deviation. Moreover, it takes into consideration obstacle size when assessing trajectory safety, further enhancing overall performance. By minimizing this cost function, the algorithm selects the optimal trajectory that adheres to constraints while providing the most

favorable driving experience. Experimental results illustrate that distinct cost functions result in different optimal trajectories.

Comparatively, with respect to the reference method outlined in [100], the trajectories planned for straight roads, curved roads, intersections, and U-shaped roads utilizing our proposed method reliably evade obstacles and actively readjust to the road's center line. Additionally, there are notable improvements in the mean values of $J_{\text{jerk}}$, namely a 13.47% reduction on straight roads, a 32.19% reduction on curved roads, a 59.36% reduction at intersections, and an 18.60% reduction on U-shaped roads. These improvements highlight the enhanced driving comfort offered by our trajectory planning method. Moreover, the mean values of $J_{\text{offset}}$ see substantial reductions of 63.72%, 13.86%, 44.36%, and 45.56% on straight roads, curved roads, intersections, and U-shaped roads, respectively, underlining our algorithm's capacity to efficiently guide a vehicle back to the road's center line following obstacle avoidance.

For obstacle avoidance scenarios with dense static obstacles, in Chapter 5, we introduced a novel algorithm for path planning comprising three key components: spatial sampling, coarse path generation, and coarse path smoothing. During the spatial sampling stage, we proposed an innovative adaptive sampling approach. This method initially constructs a three-dimensional space-aware profiling map to evaluate the current road scene. Subsequently, based on the distribution of obstacles along the road's longitudinal direction and the corresponding cost values of obstacles in the three-dimensional space-aware profiling map in the lateral direction, it generates lateral and longitudinal sampling waypoints, achieving the discretization of the road scene.

In the coarse path generation stage, the defined cost function primarily considers the smoothness of the planned path, the resistance to deviation from the road center line, and the safety of the navigable paths. To liberate the optimal path from the representation constraints of the curve model and smooth the coarse path, we employ the Taylor series to reconstruct the road model and quadratic programming techniques for coarse path smoothing.

Experimental results demonstrate that the proposed adaptive sampling algorithm dynamically adjusts the sampling strategy of waypoints based on the

quantity and distribution of obstacles within the road scenario. As obstacles increase, the number of sampled waypoints and generated paths increases. Moreover, the generated paths exhibit fewer collision paths, showcasing higher sampling effectiveness. This algorithm effectively provides a comprehensive and efficient solution set for subsequent coarse path generation.

Furthermore, when the sizes of obstacles are the same, compared to conventional sampling methods, despite a slight 2.96% increase in running time, the generated paths exhibit enhanced smoothness, more rational obstacle avoidance strategies, and a reduction of 3.48% in path length along with an 18.68% decrease in the maximum heading angle. In scenarios with different obstacle sizes, although the running time increases by 4.40%, the path length is shortened by 8.42%, and the maximum orientation angle is significantly reduced by 30.21%. The generated optimal path shows better economy, efficiency, and smoothness.

For dynamic lane-changing scenarios, in Chapter 6, we presented a novel trajectory planning algorithm under the PVD framework. In the realm of path planning, an extensive comparative analysis was conducted on the performance of Dubins curves, Sine curves, Bezier curves, and B-spline curves for lane-changing scenarios. Ultimately, the B-spline curve was chosen as the optimal lane-changing path model. This selection ensures that the planned path exhibits zero curvature at both the initiation and termination points, thereby preventing the ego vehicle from deviating from the lane center following a lane change.

For speed planning, we introduced a new method to expedite the generation of the convex space. This method dynamically creates a convex space based on the current kinematic characteristics of the ego vehicle and the prevailing traffic scenario. In contrast to the conventional DP&QP method, the proposed algorithm attains a significantly shorter average run time, realizing a notable reduction of 32.7%.

### 7.1.1 Limitations and Future Work

During the course of our research, we encountered several limitations, each of which we consider to be potential opportunities for future research. These limitations are detailed below.

For adaptive cruising scenarios on urban roads, the algorithm presented is specifically tailored to address the intricacies of static obstacle avoidance in the context of path planning, with a notable exclusion of dynamic obstacle considerations. This deliberate focus on static obstacles has provided valuable insights into establishing a robust foundation for path planning scenarios characterised by stationary obstacles. However, the lack of consideration of dynamic obstacles highlights a critical limitation in the applicability of the algorithm to real-world scenarios where the environment is subject to continuous change.

In light of this, our future research efforts will focus on addressing the challenges posed by dynamic obstacles. Recognising the dynamic nature of real-world environments, we aim to enhance the algorithm's ability to skilfully navigate through scenarios where obstacles are subject to movement. By incorporating dynamic obstacle avoidance strategies, we expect to enrich the algorithm's adaptability, thereby enhancing its utility in dynamic and ever-changing road scenarios.

For dense obstacle avoidance scenarios, our strategic vision is to improve the spatial sampling effectiveness within the adaptive path planning method. The core objective is to refine the efficiency and smoothness of the optimal paths by improving the sampling effectiveness. It is noteworthy that while this optimisation contributes to improved path quality, there is a corresponding increase in algorithmic execution time.

The key point is the potential scenario where the sampling efficiency reaches perfection at 100%. In such a state, the need for the path collision check step becomes obsolete, as paths generated with 100% sampling efficiency are inherently collision-free. This strategic decision promises significant savings in computing resources, as the path collision check step can be omitted. The resulting reduction in running time not only streamlines the algorithmic process, but also improves the real-time performance of the algorithm.

For lane-changing scenario, our current speed planning method, tailored for scenarios involving a single dynamic obstacle, has demonstrated commendable real-time performance. However, the inherent complexity of real-world traffic scenarios extends beyond single obstacles, necessitating further exploration for applications in scenarios involving multiple dynamic obstacles.

Our primary future research focus will be to seamlessly extend the existing method to address the challenges posed by multi-obstacle scenarios. The objective is to adapt and refine the current methodology to effectively address the complexities introduced by multiple dynamic obstacles during speed planning. This strategic effort aims to provide a robust solution that not only maintains its superior real-time performance, but also effectively navigates the increased complexity of diverse traffic situations commonly encountered in real-world driving environments.

# Bibliography

[1] Laurène Claussmann, Marc Revilloud, Dominique Gruyer, and Sébastien Glaser. "A Review of Motion Planning for Highway Autonomous Driving". In: *IEEE Transactions on Intelligent Transportation Systems* 21.5 (2020), pp. 1826–1848 (cit. on p. 1).

[2] Johannes Betz, Hongrui Zheng, Alexander Liniger, et al. "Autonomous vehicles on the edge: A survey on autonomous vehicle racing". In: *IEEE Open Journal of Intelligent Transportation Systems* 3 (2022), pp. 458–488 (cit. on p. 1).

[3] Wu Wen, Dongliang Xu, and Yang Xia. "A novel traffic optimization method using GRU based deep neural network for the IoV system". In: *PeerJ Computer Science* 9 (2023), e1411 (cit. on p. 1).

[4] Jiadai Wang, Jiajia Liu, and Nei Kato. "Networking and Communications in Autonomous Driving: A Survey". In: *IEEE Communications Surveys & Tutorials* 21.2 (2019), pp. 1243–1274 (cit. on p. 1).

[5] Liangkai Liu, Sidi Lu, Ren Zhong, et al. "Computing Systems for Autonomous Driving: State of the Art and Challenges". In: *IEEE Internet of Things Journal* 8.8 (2021), pp. 6469–6486 (cit. on p. 1).

[6] On-Road Automated Driving (ORAD) Committee. *Taxonomy and definitions for terms related to on-road motor vehicle automated driving systems*. SAE International, 2014 (cit. on p. 2).

[7] Scott Drew Pendleton, Hans Andersen, Xinxin Du, et al. "Perception, Planning, Control, and Coordination for Autonomous Vehicles". In: *Machines* 5.1 (2017) (cit. on p. 2).

[8] Shaoshan Liu, Liyun Li, Jie Tang, Shuang Wu, and Jean-Luc Gaudiot. *Creating autonomous vehicle systems*. Springer, 2018 (cit. on p. 2).

[9] Davy Neven, Bert De Brabandere, Stamatios Georgoulis, Marc Proesmans, and Luc Van Gool. "Towards End-to-End Lane Detection: an Instance Segmentation Approach". In: *Proc. IEEE Intelligent Vehicles Symposium (IV)*. 2018, pp. 286–291 (cit. on p. 2).

[10] Huachun Tan, Yang Zhou, Yong Zhu, Danya Yao, and Keqiang Li. “A novel curve lane detection based on Improved River Flow and RANSA”. In: *Proc. 17th International IEEE Conference on Intelligent Transportation Systems (ITSC)*. 2014, pp. 133–138 (cit. on p. 2).

[11] Wantanee Viriyasitavat, Mate Boban, Hsin-Mu Tsai, and Athanasios Vasilakos. “Vehicular Communications: Survey and Challenges of Channel and Propagation Models”. In: *IEEE Vehicular Technology Magazine* 10.2 (2015), pp. 55–66 (cit. on p. 2).

[12] Rico Krueger, Taha H. Rashidi, and Vinayak V. Dixit. “Autonomous driving and residential location preferences: Evidence from a stated choice survey”. In: *Transportation Research Part C: Emerging Technologies* 108 (2019), pp. 255–268 (cit. on p. 2).

[13] Peng Hang, Chen Lv, Yang Xing, Chao Huang, and Zhongxu Hu. “Human-Like Decision Making for Autonomous Driving: A Noncooperative Game Theoretic Approach”. In: *IEEE Transactions on Intelligent Transportation Systems* 22.4 (2021), pp. 2076–2087 (cit. on p. 2).

[14] Carl-Johan Hoel, Katherine Driggs-Campbell, Krister Wolff, Leo Laine, and Mykel J. Kochenderfer. “Combining Planning and Deep Reinforcement Learning in Tactical Decision Making for Autonomous Driving”. In: *IEEE Transactions on Intelligent Vehicles* 5.2 (2020), pp. 294–305 (cit. on p. 2).

[15] Keonyup Chu, Minchae Lee, and Myoungho Sunwoo. “Local Path Planning for Off-Road Autonomous Driving With Avoidance of Static Obstacles”. In: *IEEE Transactions on Intelligent Transportation Systems* 13.4 (2012), pp. 1599–1616 (cit. on p. 2).

[16] Bing Lu, Guofa Li, Huilong Yu, et al. “Adaptive Potential Field-Based Path Planning for Complex Autonomous Driving Scenarios”. In: *IEEE Access* 8 (2020), pp. 225294–225305 (cit. on p. 2).

[17] Siyu Teng, Xuemin Hu, Peng Deng, et al. “Motion Planning for Autonomous Driving: The State of the Art and Future Perspectives”. In: *IEEE Transactions on Intelligent Vehicles* (2023). doi: 10.1109/TIV.2023.3274536, pp. 1–21 (cit. on p. 3).

[18] Zlatan Ajanović, Enrico Regolin, Barys Shyrokau, et al. “Search-based task and motion planning for hybrid systems: Agile autonomous vehicles”. In: *Engineering Applications of Artificial Intelligence* 121 (2023), p. 105893 (cit. on p. 3).

[19] Van-Dung Hoang, Danilo Cáceres Hernández, Joko Hariyono, and Kang-Hyun Jo. “Global path planning for unmanned ground vehicle based on road map images”. In: *Proc. 7th International Conference on Human System Interactions (HSI)*. Costa da Caparica, Portugal, 2014, pp. 82–87 (cit. on p. 3).

[20] Pablo Marin-Plaza, Ahmed Hussein, David Martin, and Arturo de la Escalera. “Global and local path planning study in a ROS-based research platform for autonomous vehicles”. In: *Journal of Advanced Transportation* 2018 (2018), pp. 1–10 (cit. on p. 3).

[21] Brian Paden, Michal Čáp, Sze Zheng Yong, Dmitry Yershov, and Emilio Frazzoli. “A survey of motion planning and control techniques for self-driving urban vehicles”. In: *IEEE Transactions on Intelligent Vehicles* 1.1 (2016), pp. 33–55 (cit. on p. 3).

[22] José Ricardo Sánchez-Ibáñez, Carlos J. Pérez-del Pulgar, and Alfonso García-Cerezo. “Path Planning for Autonomous Mobile Robots: A Review”. In: *Sensors* 21.23 (2021). doi: 10.3390/s21237898 (cit. on p. 3).

[23] Vasundhara Jain, Uli Kolbe, Gabi Breuel, and Christoph Stiller. “Collision Avoidance for Multiple Static Obstacles using Path-Velocity Decomposition”. In: *IFAC-PapersOnLine* 52.8 (2019). 10th IFAC Symposium on Intelligent Autonomous Vehicles IAV 2019, pp. 265–270 (cit. on p. 4).

[24] Ding Li, Qichao Zhang, Zhongpu Xia, et al. “Planning-Inspired Hierarchical Trajectory Prediction Via Lateral-Longitudinal Decomposition for Autonomous Driving”. In: *IEEE Transactions on Intelligent Vehicles* (2023) (cit. on p. 4).

[25] Bai Li, Youmin Zhang, Zhijiang Shao, and Ning Jia. “Simultaneous versus joint computing: A case study of multi-vehicle parking motion planning”. In: *Journal of Computational Science* 20 (2017), pp. 30–40 (cit. on p. 5).

[26] Jianyu Huang, Zuguang He, Yutaka Arakawa, and Billy Dawton. “Trajectory Planning in Frenet Frame via Multi-Objective Optimization”. In: *IEEE Access* (2023) (cit. on p. 5).

[27] Omveer Sharma, N C Sahoo, and N B Puhan. “A Survey on Smooth Path Generation Techniques for Nonholonomic Autonomous Vehicle Systems”. In: *Proc. 45th Annual Conference of the IEEE Industrial Electronics Society (IES)*. Vol. 1. Lisbon, Portugal, 2019, pp. 5167–5172 (cit. on p. 11).

[28] O. Khatib. “Real-time obstacle avoidance for manipulators and mobile robots”. In: *Proc. IEEE International Conference on Robotics and Automation (ICRA)*. Vol. 2. St. Louis, MO, USA, 1985, pp. 500–505 (cit. on p. 11).

[29] Ulises Orozco-Rosas, Oscar Montiel, and Roberto Sepúlveda. “Mobile robot path planning using membrane evolutionary artificial potential field”. In: *Applied Soft Computing* 77 (2019), pp. 236–251 (cit. on p. 12).

[30] Bing Lu, Hongwen He, Huilong Yu, et al. “Hybrid path planning combining potential field with sigmoid curve for autonomous driving”. In: *Sensors* 20.24 (2020), p. 7197 (cit. on p. 12).

[31] Yadollah Rasekhipour, Amir Khajepour, Shih-Ken Chen, and Bakhtiar Litkouhi. “A Potential Field-Based Model Predictive Path-Planning Controller for Autonomous Road Vehicles”. In: *IEEE Transactions on Intelligent Transportation Systems* 18.5 (2017), pp. 1255–1267 (cit. on p. 12).

[32] Yanjun Huang, Haitao Ding, Yubiao Zhang, et al. “A Motion Planning and Tracking Framework for Autonomous Vehicles Based on Artificial Potential Field Elaborated Resistance Network Approach”. In: *IEEE Transactions on Industrial Electronics* 67.2 (2020), pp. 1376–1386 (cit. on p. 12).

[33] Mihail Pivtoraiko, Ross A Knepper, and Alonzo Kelly. “Differentially constrained mobile robot motion planning in state lattices”. In: *Journal of Field Robotics* 26.3 (2009), pp. 308–333 (cit. on p. 12).

[34] Maxim Likhachev and Dave Ferguson. “Planning long dynamically feasible maneuvers for autonomous vehicles”. In: *The International Journal of Robotics Research* 28.8 (2009), pp. 933–945 (cit. on p. 13).

[35] Aleksandr Kushleyev and Maxim Likhachev. “Time-bounded lattice for efficient planning in dynamic environments”. In: *Proc. IEEE International Conference on Robotics and Automation (ICRA)*. Kobe, Japan, 2009, pp. 1662–1668 (cit. on p. 13).

[36] Matthew McNaughton, Chris Urmson, John M. Dolan, and Jin-Woo Lee. “Motion planning for autonomous driving with a conformal spatiotemporal lattice”. In: *Proc. IEEE International Conference on Robotics and Automation (ICRA)*. Shanghai, China, 2011, pp. 4889–4895 (cit. on p. 13).

[37] Edsger W Dijkstra. “A note on two problems in connexion with graphs”. In: *Edsger Wybe Dijkstra: His Life, Work, and Legacy*. 2022, pp. 287–290 (cit. on p. 13).

[38] Peter E. Hart, Nils J. Nilsson, and Bertram Raphael. “A Formal Basis for the Heuristic Determination of Minimum Cost Paths”. In: *IEEE Transactions on Systems Science and Cybernetics* 4.2 (1968), pp. 100–107 (cit. on p. 13).

[39] Dmitri Dolgov, Sebastian Thrun, Michael Montemerlo, and James Diebel. “Practical search techniques in path planning for autonomous driving”. In: *Ann Arbor* 1001.48105 (2008), pp. 18–80 (cit. on p. 13).

[40] A. Stentz. “Optimal and efficient path planning for partially-known environments”. In: *Proc. IEEE International Conference on Robotics and Automation (ICRA)*. San Diego, CA, USA, 1994, 3310–3317 vol.4 (cit. on p. 13).

[41] Maxim Likhachev, Dave Ferguson, Geoff Gordon, Anthony Stentz, and Sebastian Thrun. “Anytime Dynamic A*: An Anytime, Replanning Algorithm”. In: *Proc. 15th International Conference on International Conference on Automated Planning and Scheduling (ICAPS)*. Monterey, California, USA, 2005, 262–271 (cit. on p. 14).

[42] Sven Koenig and Maxim Likhachev. “D*lite”. In: *Proc. 18th National Conference on Artificial Intelligence (AAAI-02)*. Edmonton, Alberta, Canada: American Association for Artificial Intelligence, 2002, 476–483 (cit. on p. 14).

[43] Sertac Karaman and Emilio Frazzoli. “Sampling-based algorithms for optimal motion planning”. In: *The International Journal of Robotics Research* 30.7 (2011), pp. 846–894 (cit. on p. 14).

[44] Bai Li, Qi Kong, Youmin Zhang, et al. “On-road Trajectory Planning with Spatio-temporal RRT* and Always-feasible Quadratic Program”. In: *Proc. IEEE 16th International Conference on Automation Science and Engineering (CASE)*. Hongkong, China, 2020, pp. 942–947 (cit. on p. 14).

[45] J.J. Kuffner and S.M. LaValle. “RRT-connect: An efficient approach to single-query path planning”. In: *Proc. IEEE International Conference on Robotics and Automation (ICRA)*. Vol. 2. San Francisco, CA, USA, 2000, pp. 995–1001 (cit. on p. 14).

[46] Steven M. LaValle and Jr. James J. Kuffner. “Randomized Kinodynamic Planning”. In: *The International Journal of Robotics Research* 20.5 (2001), pp. 378–400 (cit. on p. 14).

[47] J.J. Kuffner and S.M. LaValle. “RRT-connect: An efficient approach to single-query path planning”. In: *Proc. IEEE International Conference on Robotics and Automation (ICRA)*. Vol. 2. 2000, 995–1001 vol.2 (cit. on p. 14).

[48] Binghui Li and Badong Chen. “An Adaptive Rapidly-Exploring Random Tree”. In: *IEEE/CAA Journal of Automatica Sinica* 9.2 (2022), pp. 283–294 (cit. on p. 15).

[49] Chunlai Feng. “Research on Autonomous Vehicle Motion Planning Method Using Parameterized RRT Based on Guiding Area”. PhD thesis. University of Science and Technology of China, Jan. 2017 (cit. on p. 15).

[50] Xuemin Hu, Long Chen, Bo Tang, Dongpu Cao, and Haibo He. “Dynamic path planning for autonomous driving on various roads with avoidance of static and moving obstacles”. In: *Mechanical Systems and Signal Processing* 100 (2018), pp. 482–500 (cit. on p. 15).

[51] Moritz Werling, Julius Ziegler, Sören Kammel, and Sebastian Thrun. “Optimal trajectory generation for dynamic street scenarios in a frenet frame”. In: *Proc. IEEE International Conference on Robotics and Automation (ICRA)*. 2010, pp. 987–993 (cit. on p. 16).

[52] Haoyang Fan, Fan Zhu, Changchun Liu, et al. *Baidu apollo em motion planner*. *arXiv:1807.08048*. 2018 (cit. on p. 16).

[53] Yajia Zhang, Hongyi Sun, Jinyun Zhou, et al. “Optimal Vehicle Path Planning Using Quadratic Optimization for Baidu Apollo Open Platform”. In: *Proc. IEEE Intelligent Vehicles Symposium (IV)*. Las Vegas, NV, USA, 2020, pp. 978–984 (cit. on p. 16).

[54] Jinyun Zhou, Runxin He, Yu Wang, et al. “Autonomous Driving Trajectory Optimization With Dual-Loop Iterative Anchoring Path Smoothing and Piecewise-Jerk Speed Optimization”. In: *IEEE Robotics and Automation Letters* 6.2 (2021), pp. 439–446 (cit. on p. 16).

[55] Wonteak Lim, Seongjin Lee, Myoungho Sunwoo, and Kichun Jo. “Hybrid Trajectory Planning for Autonomous Driving in On-Road Dynamic Scenarios”. In: *IEEE Transactions on Intelligent Transportation Systems* 22.1 (2021), pp. 341–355 (cit. on p. 16).

[56] Siyu Teng, Long Chen, Yunfeng Ai, et al. “Hierarchical Interpretable Imitation Learning for End-to-End Autonomous Driving”. In: *IEEE Transactions on Intelligent Vehicles* 8.1 (2023). doi: 10.1109/TIV.2022.3225340, pp. 673–683 (cit. on p. 17).

[57] Yihan Hu, Jiazhi Yang, Li Chen, et al. “Planning-oriented Autonomous Driving”. In: *Proc. the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2023 (cit. on p. 17).

[58] Fei Ye, Shen Zhang, Pin Wang, and Ching-Yao Chan. “A Survey of Deep Reinforcement Learning Algorithms for Motion Planning and Control of Autonomous Vehicles”. In: *Proc. IEEE Intelligent Vehicles Symposium (IV)*. Nagoya, Japan, 2021, pp. 1073–1080 (cit. on p. 17).

[59] Luc Le Mero, Dewei Yi, Mehrdad Dianati, and Alexandros Mouzakitis. “A Survey on Imitation Learning Techniques for End-to-End Autonomous Vehicles”. In: *IEEE Transactions on Intelligent Transportation Systems* 23.9 (2022). doi: 10.1109/TITS.2022.3144867, pp. 14128–14147 (cit. on p. 17).

[60]Yuchuan Du, Jing Chen, Cong Zhao, et al. “Comfortable and energy-efficient speed control of autonomous vehicles on rough pavements using deep reinforcement learning”. In: *Transportation Research Part C: Emerging Technologies* 134 (2022), p. 103489 (cit. on p. 17).

[61]Szilárd Aradi. “Survey of Deep Reinforcement Learning for Motion Planning of Autonomous Vehicles”. In: *IEEE Transactions on Intelligent Transportation Systems* 23.2 (2022). doi: 10.1109/TITS.2020.3024655, pp. 740–759 (cit. on p. 17).

[62]Dean A Pomerleau. “Alvinn: An autonomous land vehicle in a neural network”. In: *Advances in neural information processing systems* 1 (1988) (cit. on p. 17).

[63]Davi Frossard and Raquel Urtasun. “End-to-end Learning of Multi-sensor 3D Tracking by Detection”. In: *Proc. IEEE International Conference on Robotics and Automation (ICRA)*. Brisbane, QLD, Australia, 2018, pp. 635–642 (cit. on p. 17).

[64]Stephane Ross, Geoffrey Gordon, and Drew Bagnell. “A Reduction of Imitation Learning and Structured Prediction to No-Regret Online Learning”. In: *Proc. 14th International Conference on Artificial Intelligence and Statistics (AISTATS)*. Ed. by Geoffrey Gordon, David Dunson, and Miroslav Dudík. Vol. 15. Proceedings of Machine Learning Research. Fort Lauderdale, FL, USA, 2011, pp. 627–635 (cit. on p. 18).

[65]He He, Jason Eisner, and Hal Daume. “Imitation Learning by Coaching”. In: *Proc. Advances in Neural Information Processing Systems (NeurIPS)*. Ed. by F. Pereira, C.J. Burges, L. Bottou, and K.Q. Weinberger. Vol. 25. Curran Associates, Inc., 2012 (cit. on p. 18).

[66]Felipe Codevilla, Matthias Müller, Antonio López, Vladlen Koltun, and Alexey Dosovitskiy. “End-to-end driving via conditional imitation learning”. In: *Proc. IEEE International Conference on Robotics and Automation (ICRA)*. Brisbane, QLD, Australia, 2018, pp. 4693–4700 (cit. on p. 18).

[67]B Ravi Kiran, Ibrahim Sobh, Victor Talpaert, et al. “Deep reinforcement learning for autonomous driving: A survey”. In: *IEEE Transactions on Intelligent Transportation Systems* 23.6 (2021), pp. 4909–4926 (cit. on p. 18).

[68]Ahmad EL Sallab, Mohammed Abdou, Etienne Perot, and Senthil Yogamani. “Deep reinforcement learning framework for autonomous driving”. In: *arXiv preprint arXiv:1704.02532* (2017) (cit. on p. 19).

[69] Jianyu Chen, Shengbo Eben Li, and Masayoshi Tomizuka. “Interpretable end-to-end urban autonomous driving with latent deep reinforcement learning”. In: *IEEE Transactions on Intelligent Transportation Systems* 23.6 (2021), pp. 5068–5078 (cit. on p. 19).

[70] Timothy P Lillicrap, Jonathan J Hunt, Alexander Pritzel, et al. “Continuous control with deep reinforcement learning”. In: *arXiv preprint arXiv:1509.02971* (2015) (cit. on p. 19).

[71] Kai Yang, Xiaolin Tang, Sen Qiu, et al. “Towards robust decision-making for autonomous driving on highway”. In: *IEEE Transactions on Vehicular Technology* (2023) (cit. on p. 19).

[72] Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, et al. “Asynchronous Methods for Deep Reinforcement Learning”. In: *Pro. 33rd International Conference on Machine Learning (ICML)*. Ed. by Maria Florina Balcan and Kilian Q. Weinberger. Vol. 48. Proceedings of Machine Learning Research. New York, New York, USA, 2016, pp. 1928–1937 (cit. on p. 19).

[73] Marin Toromanoff, Emilie Wirbel, and Fabien Moutarde. “End-to-End Model-Free Reinforcement Learning for Urban Driving Using Implicit Affordances”. In: *Proc. the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2020 (cit. on p. 19).

[74] Xiaodan Liang, Tairui Wang, Luona Yang, and Eric Xing. “CIRL: Controllable Imitative Reinforcement Learning for Vision-Based Self-Driving”. In: *Proc. The European Conference on Computer Vision (ECCA)*. Berlin, Heidelberg, Germany, 2018, 604–620 (cit. on p. 19).

[75] Yang Yang, Li Juntao, and Peng Lingling. “Multi-robot path planning based on a deep reinforcement learning DQN algorithm”. In: *CAAI Transactions on Intelligence Technology* 5.3 (2020), pp. 177–183 (cit. on p. 19).

[76] Zuxin Liu, Baiming Chen, Hongyi Zhou, et al. “Mapper: Multi-agent path planning with evolutionary reinforcement learning in mixed dynamic environments”. In: *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS*. IEEE. 2020, pp. 11748–11754 (cit. on p. 19).

[77] Jason Kong, Mark Pfeiffer, Georg Schildbach, and Francesco Borrelli. “Kinematic and dynamic vehicle models for autonomous driving control design”. In: *Proc. IEEE Intelligent Vehicles Symposium (IV)*. IEEE. 2015, pp. 1094–1099 (cit. on p. 22).

[78] Luqi Tang, Fuwu Yan, Bin Zou, Kewei Wang, and Chen Lv. “An improved kinematic model predictive control for high-speed path tracking of autonomous vehicles”. In: *IEEE Access* 8 (2020), pp. 51400–51413 (cit. on p. 22).

[79]Hormoz Marzbani, Hamid Khayyam, Đai Võ Quoc, and Reza N Jazar. “Autonomous vehicles: Autodriver algorithm and vehicle dynamics”. In: *IEEE Transactions on Vehicular Technology* 68.4 (2019), pp. 3201–3211 (cit. on p. 23).

[80]Dongfang Dang, Feng Gao, and Qiuxia Hu. “Motion planning for autonomous vehicles considering longitudinal and lateral dynamics coupling”. In: *Applied Sciences* 10.9 (2020), p. 3180 (cit. on p. 23).

[81]Philip Polack, Florent Altché, Brigitte d’Andréa Novel, and Arnaud de La Fortelle. “The kinematic bicycle model: A consistent model for planning feasible trajectories for autonomous vehicles?” In: *Proc. IEEE Intelligent Vehicles Symposium (IV)*. IEEE. 2017, pp. 812–818 (cit. on p. 23).

[82]Florent Altché, Philip Polack, and Arnaud de La Fortelle. “High-speed trajectory planning for autonomous vehicles using a simple dynamic model”. In: *Proc. 20th IEEE Intelligent Transportation Systems Conference (ITSC)*. IEEE. 2017, pp. 1–7 (cit. on p. 24).

[83]Bai Li, Yakun Ouyang, Li Li, and Youmin Zhang. “Autonomous driving on curvy roads without reliance on frenet frame: A cartesian-based trajectory planning method”. In: *IEEE Transactions on Intelligent Transportation Systems* 23.9 (2022), pp. 15729–15741 (cit. on p. 25).

[84]Yu Zhang, Huiyan Chen, Steven L Waslander, et al. “Hybrid trajectory planning for autonomous driving in highly constrained environments”. In: *IEEE Access* 6 (2018), pp. 32800–32819 (cit. on p. 25).

[85]Yinghui Wang and Zhen Lin. “Research on path planning for autonomous vehicle based on Frenet system”. In: *Journal of Engineering Research* (2023), p. 100080 (cit. on p. 25).

[86]Bai Li and Youmin Zhang. “Fast trajectory planning in Cartesian rather than Frenet frame: A precise solution for autonomous driving in complex urban scenarios”. In: *IFAC-PapersOnLine* 53.2 (2020), pp. 17065–17070 (cit. on p. 26).

[87]Stefanie Manzinger, Christian Pek, and Matthias Althoff. “Using reachable sets for trajectory planning of automated vehicles”. In: *IEEE Transactions on Intelligent Vehicles* 6.2 (2020), pp. 232–248 (cit. on p. 26).

[88]Federico Milano, Georgios Tzounas, Ioannis Dassios, and Taulant Kerci. “Applications of the frenet frame to electric circuits”. In: *IEEE Transactions on Circuits and Systems I: Regular Papers* 69.4 (2021), pp. 1668–1680 (cit. on p. 27).

[89] Jiajia Chen, Rui Zhang, Wei Han, et al. “Path planning for autonomous vehicle based on a Two-Layered planning model in complex environment”. In: *Journal of Advanced Transportation* 2020 (2020), pp. 1–14 (cit. on p. 27).

[90] Wenda Xu, Jia Pan, Junqing Wei, and John M. Dolan. “Motion planning under uncertainty for on-road autonomous driving”. In: *Proc. IEEE International Conference on Robotics and Automation (ICRA)*. Hong Kong, China, 2014, pp. 2507–2512 (cit. on p. 32).

[91] Xiangjun Qian, Florent Altché, Philipp Bender, Christoph Stiller, and Arnaud de La Fortelle. “Optimal trajectory planning for autonomous driving integrating logical constraints: An MIQP perspective”. In: *Proc. 19th IEEE Intelligent Transportation Systems Conference (ITSC)*. IEEE. 2016, pp. 205–210 (cit. on p. 33).

[92] Dasol Jeong and Seibum B Choi. “Efficient Trajectory Planning for Autonomous Vehicles Using Quadratic Programming with Weak Duality”. In: *IEEE Transactions on Intelligent Vehicles* (2023) (cit. on p. 33).

[93] Han Li, Peng Chen, Guizhen Yu, et al. “Trajectory Planning for Autonomous Driving in Unstructured Scenarios Based on Deep Learning and Quadratic Optimization”. In: *IEEE Transactions on Vehicular Technology* (2023) (cit. on p. 33).

[94] Dave Ferguson, Thomas M Howard, and Maxim Likhachev. “Motion planning in urban environments”. In: *Journal of Field Robotics* 25.11-12 (2008), pp. 939–960 (cit. on p. 35).

[95] Xiaohui Li, Zhenping Sun, Dongpu Cao, Zhen He, and Qi Zhu. “Real-time trajectory planning for autonomous urban driving: Framework, algorithms, and verifications”. In: *IEEE/ASME Transactions on mechatronics* 21.2 (2015), pp. 740–753 (cit. on p. 35).

[96] Zhi Cai, Xuerui Cui, Xing Su, et al. “A novel vector-based dynamic path planning method in urban road network”. In: *IEEE Access* 8 (2019), pp. 9046–9060 (cit. on p. 35).

[97] Longhao Yan, Ping Wang, Jingwen Yang, et al. “Refined path planning for emergency rescue vehicles on congested urban arterial roads via reinforcement learning approach”. In: *Journal of Advanced Transportation* 2021 (2021), pp. 1–12 (cit. on p. 35).

[98] Levent Guvenc, Bilin Aksun-Guvenc, Sheng Zhu, and Sukru Yaren Gelbal. *Autonomous Road Vehicle Path Planning and Tracking Control*. John Wiley & Sons, 2021 (cit. on p. 36).

[99] Carlotta Giannelli, Duccio Mugnaini, and Alessandra Sestini. “Path planning with obstacle avoidance by G1 PH quintic splines”. In: *Computer-Aided Design* 75 (2016), pp. 47–60 (cit. on p. 42).

[100] Minxiangy WEI, Decheng TENG, and Shufan WU. “Trajectory planning and optimization algorithm for automated driving based on Frenet coordinate system”. In: *Control and Decision* 36.4 (2021), pp. 815–824 (cit. on pp. 47, 50, 51, 55, 110).

[101] Julian Bock, Robert Krajewski, Tobias Moers, et al. “The inD Dataset: A Drone Dataset of Naturalistic Road User Trajectories at German Intersections”. In: *Proc. IEEE Intelligent Vehicles Symposium (IV)*. Las Vegas, NV, USA, 2020, pp. 1929–1934 (cit. on p. 49).

[102] Wei Zhan, Liting Sun, Di Wang, et al. *INTERACTION Dataset: An INTERnational, Adversarial and Cooperative moTION Dataset in Interactive Driving Scenarios with Semantic Maps*. *arXiv: 1910.03088*. 2019 (cit. on p. 49).

[103] Chengyuan Ma, Chunhui Yu, and Xiaoguang Yang. “Trajectory planning for connected and automated vehicles at isolated signalized intersections under mixed traffic environment”. In: *Transportation research part C: emerging technologies* 130 (2021). doi: 10.1016/j.trc.2021.103309, p. 103309 (cit. on p. 49).

[104] Wonteak Lim, Seongjin Lee, Myoungho Sunwoo, and Kichun Jo. “Hierarchical Trajectory Planning of an Autonomous Car Based on the Integration of a Sampling and an Optimization Method”. In: *IEEE Transactions on Intelligent Transportation Systems* 19.2 (2018), pp. 613–626 (cit. on p. 80).

[105] Kamal Kant and Steven W Zucker. “Toward efficient trajectory planning: The path-velocity decomposition”. In: *The International Journal of Robotics Research* 5.3 (1986), pp. 72–89 (cit. on p. 90).

[106] Qiongqiong Li, Yiqi Xu, Shengqiang Bu, and Jiafu Yang. “Smart vehicle path planning based on modified PRM algorithm”. In: *Sensors* 22.17 (2022), p. 6581 (cit. on p. 91).

[107] Vahide Bulut. “Path planning for autonomous ground vehicles based on quintic trigonometric Bézier curve: Path planning based on quintic trigonometric Bézier curve”. In: *Journal of the Brazilian Society of Mechanical Sciences and Engineering* 43 (Feb. 2021) (cit. on p. 92).

[108] Wenyu Cai, Meiyan Zhang, and Yahong Rosa Zheng. “Task assignment and path planning for multiple autonomous underwater vehicles using 3D dubins curves”. In: *Sensors* 17.7 (2017), p. 1607 (cit. on p. 93).

[109] Jinyun Zhou, Runxin He, Yu Wang, et al. “Autonomous Driving Trajectory Optimization With Dual-Loop Iterative Anchoring Path Smoothing and Piecewise-Jerk Speed Optimization”. In: *IEEE Robotics and Automation Letters* 6.2 (2021), pp. 439–446 (cit. on p. 103).

# Appendix

## Publish Works

### Journal

1. J. Huang, Z. He, Y. Arakawa and B. Dawton, "Trajectory Planning in Frenet Frame via Multi-Objective Optimization". In: *IEEE Access* 11 (2023), pp. 70764-70777. doi: 10.1109/ACCESS.2023.3294713.
2. J. Huang, B. Dawton and Y. Arakawa, "An Adaptive Path Planning Method for Curved Roads via Three-Dimensional Space-Aware Profiling Maps". In: *IEEE Access* 12 (2024), pp. 1458-1473. doi: 10.1109/ACCESS.2023.3347399.

### Conference

1. J. Huang and Y. Arakawa, "Faster Trajectory Planning for Lane Change Scenarios with Dynamic Environment". In: *Proc. 8th International Conference on Robotics, Control and Automation (ICRCA)*. Shanghai, China, 2024.