IEEE Transactions on Automation Science and Engineering (T-ASE) paper, presented at ICRA 2026, Vienna, Austria. Cite as T-ASE paper.

Tracailer: An Efficient Trajectory Planner for Tractor-Trailer Robots in Unstructured Environments

Long Xu[1,2], Kaixin Chai[2], Boyuan An[1], Shuhang Ji[1], Zhenyu Hou[2], Jiaxiang Gan[2], Qianhao Wang[1,2], Yuan Zhou[1,2], Xiaoying Li[2], Junxiao Lin[1,2], Zhichao Han[1,2], Chao Xu[1,2], Yanjun Cao[1,2], and Fei Gao[1,2]

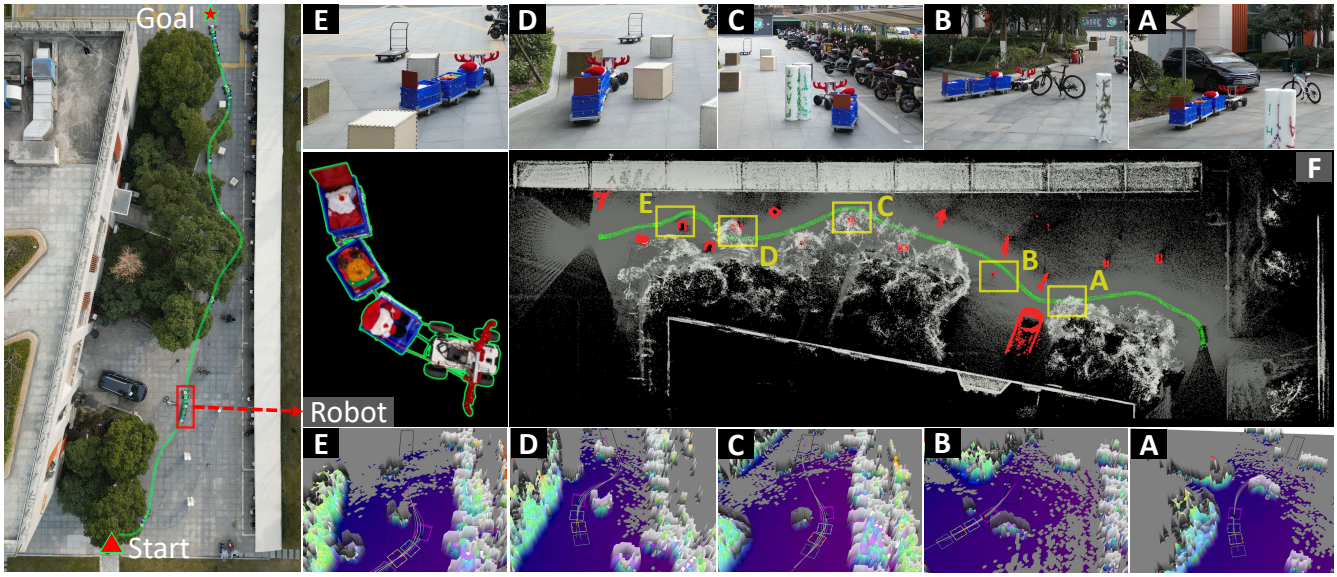


Fig. 1: A tractor-trailer robot navigating autonomously through narrow lanes. Subfigures A∼E at the top and bottom of the figure correspond to the third-person viewpoint and view from Rviz [1] at the moment of the yellow box in the Subfigure F, respectively.

***Abstract*— The tractor-trailer robot consists of a drivable tractor and one or more non-drivable trailers connected via hitches. Compared to typical car-like robots, the addition of trailers provides greater transportation capability. However, this also complicates motion planning due to the robot's complex kinematics, high-dimensional state space, and deformable structure. To efficiently plan safe, time-optimal trajectories that adhere to the kinematic constraints of the robot and address the challenges posed by its unique features, this paper introduces a lightweight, compact, and high-order smooth trajectory representation for tractor-trailer robots. Based on it, we design an efficiently solvable spatial-temporal trajectory optimization problem. To deal with deformable structures, which leads to difficulties in collision avoidance, we fully leverage the collision-free regions of the environment, directly applying deformations to trajectories in continuous space. This approach not requires constructing safe regions from the environment using convex approximations through collision-free seed points before each optimization, avoiding the loss of the solution space, thus reducing the dependency of the optimization on initial values. Moreover, a multi-terminal fast path search algorithm is proposed to generate the initial values for optimization. Extensive simulation experiments demonstrate that our approach achieves several-fold improvements in efficiency compared to existing algorithms, while also ensuring lower curvature and trajectory duration. Real-world experiments involving the transportation, loading and unloading of goods in both indoor and outdoor scenarios further validate the effectiveness of our method. The source code is accessible at https://github.com/Tracailer/Tracailer.**

***Note to Practitioners*—This paper addresses the challenges of motion planning for tractor-trailer robots, which are crucial in industries like logistics and agriculture. Our approach introduces a lightweight trajectory planning method that efficiently generates safe and time-optimal paths while considering the unique kinematic constraints of these vehicles. By utilizing the environment's collision-free areas, we enhance safety and reduce reliance on initial conditions. While our method shows significant efficiency improvements in simulations and real-world tests, its effectiveness may vary in extremely dynamic environments. Practitioners can easily adopt our open-source code to integrate this planning technique into their operations, improving the efficiency and safety of tractor-trailer systems and paving the way for future**

This work was supported by the National Key R&D Program of China under Grant No. 2023YFB4706600, the Zhejiang Provincial Science and Technology Plan Project under Grant No. 2024C01170 and the National Natural Science Foundation of China under Grant No. 62322314.

[1]State Key Laboratory of Industrial Control Technology, Zhejiang University, Hangzhou 310027, China. *Corresponding author: Fei Gao*

[2]Huzhou Institute of Zhejiang University, Huzhou 313000, China.

E-mail: {gaolon, fgaoaa}@zju.edu.cn

advancements in autonomous transportation.

Motion Planning, Trajectory Optimization, Tractor-trailer Robot, Nonholonomic Dynamics

I. Introduction

In recent years, autonomous driving has gained a lot of interest and great growth due to its potential social benefits. When transporting large cargo, attention often shifts to tractor-trailer systems, such as semi-trucks, due to their capacity to carry more through the use of trailers. Tractor-trailer robot is a class of vehicles consisting of a drivable tractor and many unpowered trailers. The trailers are often connected to the tractor or other trailers by hinges, so that the motions of the trailers can be controlled by controlling that of the tractor [2].

Motion planning is a crucial component of autonomous navigation frameworks, and tractor-trailer robots are no exception. Similarly to car-like robots, the trajectory planner for tractor-trailer robots aims to generate trajectories that minimize time consumption while satisfying kinematic constraints, enabling smooth and safe navigation to the target area. Although motion planning for car-like robots has been studied relatively extensively, these techniques cannot be directly applied to tractor-trailer robots, since it is necessary to consider not only the state of the tractor but also the state of the trailer when moving. Specifically, compared to car-like robots, the challenges in motion planning for tractor-trailer robots are mainly caused by the following three factors:

1) **Complex kinematics:** The hinge structure and the non-drivability of the trailers introduce more complex kinematic constraints. The coupling of the underactuated constraints of the trailers and the nonholonomic constraints of the tractor [3] requires the planner capable of handling more equation constraints. Meanwhile, it must avoid the jackknife, which will make the entire robot system uncontrollable [4].
2) **High-dimensional state space:** Each trailer attached to the robot introduces three additional dimensions to the robot in space, i.e. position and yaw angle, which can significantly increase the time consumed by search- or sampling-based path planning algorithms [5], [6]. Similarly, for optimization-based methods [3], [7]–[10], the increase in dimension of the state leads to an increase in the dimension of the optimization variables, which may result in more memory consumption and slower convergence.
3) **Deformable structure:** The deformability of the tractor-trailer robot, caused by its hinged structure, increases the complexity of collision avoidance. Since the shape of robot changes over time, obstacle avoidance constraints must be applied to each individual unit of the robot. This challenge becomes even more pronounced with multiple trailers, where additional vehicle-to-vehicle collision avoidance constraints must also be considered.

To cope with the first two problems, we draw inspiration from the efficiency of polynomial trajectories optimization and differential flatness [11] in motion planning for car-like robots [12], [13], proposing a new trajectory representation for tractor-trailer robots. We model the spatial-temporal trajectory optimization problem of tractor-trailer robots in continuous space, thus eliminating the discretization of the motion process and substantially improving the efficiency of the optimization. Benefiting from differential flatness [11], we can explicitly derive higher-order states of the robot and reduce the dimensionality of the optimization variables. However, the singularity of differential flatness brought about by kinematics of the tractor may lead to a decrease in the optimization success rate and smoothness of the trajectory, which may make trajectory tracking difficult. Thus, in this work, we introduce *slackened arc length* to eliminate this singularity, improving the success rate of optimization. In addition, this approach makes the feasibility constraints easier to satisfy, leading to a further enhancement in efficiency.

For the third problem of deformed structures, we fully leverage the collision-free regions of the environment, directly applying deformations to trajectories in continuous space. This approach not requires constructing safe regions from the environment using convex approximations through collision-free seed points before each optimization, avoiding the loss of the solution space, thus reducing the dependency of the optimization on initial values, making it easier for the optimizer to find a better solution. With this advantage, to further boost the efficiency, we propose a multi-terminal path search algorithm for tractor-trailer robots, which can searches only in the $SE(2)$ space. Compared to exploration in larger state spaces, our approach achieves nearly an order of magnitude improvement in efficiency with minimal loss in optimality. Contributions of this paper are:

1) We propose a new trajectory representation for tractor-trailer robots that retains the high efficiency benefits from polynomial representations while eliminating the singularity of differential flatness, further improving the success rate of the trajectory optimization.
2) Based on the above trajectory representation, we propose a new optimization-based trajectory planner (Tracailer) for **trac**tor-tr**ailer** robots. It directly applies deformations to trajectories in continuous space for ensuring safety, reducing the dependency of optimization on initial values. Meanwhile, an efficient multi-terminal path finder is designed and embedded into it for acquiring the initial values of the problem, achieving higher efficiency without losing much optimality.
3) To efficiently solve the optimization problem in the planner, we reduce the dimensionality of the optimization variables and constraints based on differential homogeneous mapping and Lagrange multipliers.
4) Moreover, we build an autonomous navigation system for tractor-trailer robots, including a real-time replanning and mapping module, demonstrating the effectiveness and efficiency of the proposed method in extensive simulations and real-world experiments, such as those shown in Fig. 1. The software is open-sourced to

promote further research in this field.

II. Related Works

Motion planning for tractor-trailer robots can be mainly classified into geometry-based methods and optimization-based methods.

A. Geometry-based Methods

Geometry-based methods usually plan only the geometric path and design the path tracking controller to track it. The geometric path is usually generated by search- or sampling-based methods. Search-based methods typically pre-construct a grid map or graph structure that represents the environment, and then perform graph searching within it to find paths [14]. For example, Liu et al. [15] utilized the conception of equivalent size for obstacles growing and robot shrinking, planning the path by applying a heuristic genetic algorithm. Sun et al. [16] built a global compound roadmap by combining the local regular roadmap with the universal probabilistic roadmap, lowering the complexity of the planning computation. To improve the efficiency and path quality, Ljungqvist et al. [5] generated a finite set of kinematically feasible motion primitives offline, searching solutions for a general 2-trailer system in a regular state lattice created by these motion primitives. They also proposed a novel parametrization of the reachable state space to make the graph-search problem tractable for real-time applications. Furthermore, by simplifying the design of motion primitives and utilizing reinforcement learning to obtain a well-defined heuristic function, Leu et al. presented an improved A-search guided tree [17] that allows quick off-lattice exploration to find a solution.

Sampling-based methods usually define a space for a path planning task. They maintain a tree structure and iteratively sample in this space, expanding tree nodes according to some criteria, ultimately extracting the trajectories from the tree containing the start and goal. By customizing the sampling space based on motion primitives instead of the space containing all $SE(2)$ states of vehicles in the tractor trailer robot, Cheng et al. [18], Lattarulo et al. [6] used the Rapidly-exploring Random Tree [19], [20] (RRT) algorithm to efficiently plan a feasible path that is satisfied with the nonholonomic and mechanical constraints. Moreover, Manav et al. [21] proposed an iterative analytical method, combining it with the Closed-Loop Rapidly Exploring Random Tree (CL-RRT) approach in cascade path planning, allowing the generation of both kinematically feasible and deterministic parking maneuvers with obstacle avoidance.

After obtaining the path by search- or sampling-based methods, the trajectory tracker will choose a piece of path as reference based on the current state of the robot and compute the control command sent to the robot [22]. For example, in work [18], the path is followed with fuzzy control and Line-of-sight approach [23]. While in work [24], a semidefinite programming (SDP) problem is constructed to combine with motion primitive to get the command. Dahlmann et al. [25], [26] proposed model predictive controllers to perform local optimizations with Voronoi field [27] for suboptimal reference trajectories.

B. Optimization-based Methods

Although geometry-based methods can work in various scenarios, these approach always search in a discrete space, where the optimality of the solution obtained is positively correlated with the duration of the search and the granularity of the space. When information of higher dimensions in the state of the robot is considered to find a more optimal solution, these methods need to explore larger space, whereupon the problem of combinatorial explosion is magnified, severely affecting the efficiency. To find better solutions in such a large space, optimization-based approaches emerged. These methods often model motion planning as an optimization problem and use numerical solvers to solve it, whose initial values often derived from geometric paths obtained by search- or sampling-based methods. Trajectory optimization typically explores optimal solutions in a continuous space, which can take into account information about higher-order states with guarantees of local optimality. Muralidhara et al. [8] modeled the motion planning problem of tractor-trailer robots as Optimal Control Problem (OCP), describing the trajectory by states in discrete timestamps, solving it by direct multiple shooting method. But in obstacle avoidance, this approach only considers not leaving the tracking line of the traffic lane, which cannot be directly applied to unstructured complex environments. Mohamed et al. [28] proposed an approach combining artificial potential fields and optimal control theory to achieve a more generalized obstacle avoidance for tractor-trailer robot. However, this method can easily fall into a local optimum when the environment is complex, preventing the robot from reaching the target region. Also using the potential field approach to represent the environment, Wang et al. [29] adopt path integral policy improvement to optimize the paths from geometry based methods, solving the problem of poor performance of graph search methods in narrow scenes the limitations by resolution and search space. Nevertheless, this approach is time-consuming, making it difficult to apply to tasks that require online replanning, such as exploring unknown environments and navigating through dynamically changing scenarios.

The method proposed in this paper is also an optimization-based algorithm. To plan efficiently, we do not adopt discrete states to represent trajectory and model the problem as an OCP, but propose a lightweight, compact and high-order smooth trajectory representation, reducing the dimension of optimization variables and simplifying the problem.

In works [9], [30], [31], both obstacles and robots are represented by many circles, as this simple form can be easily modeled into OCP. However, real-world environments are complex and cannot be accurately modeled using only circles. To represent the environment more concisely and accurately, some works [3], [10], [32]–[34] use convex polygons to approximate obstacles. They also modeled the motion planning problem for tractor-trailer robots as OCP,

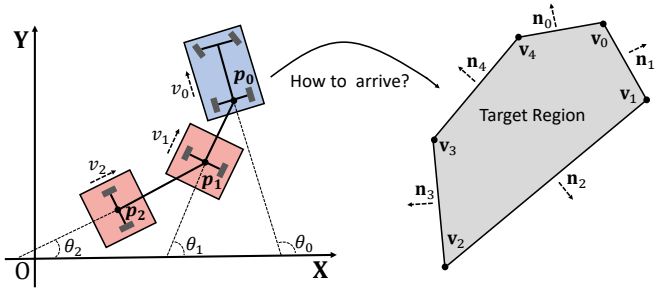


Fig. 2: An example of a tractor-trailer robot (N = 2). The first trailer is hooked to the rear center of the tractor, while the others are hooked to the center of the previous one.

considering safety constraints and duration in the optimization. Their methods require applying constraints to each obstacle and each vehicle of the robot at each moment. This feature tightly couples their computational complexity to the environment, limiting the efficiency of planning in obstacle-dense environments. To address this issue, some works [7], [35] shift to Safe Travel Corridors (STC) as obstacle avoidance constraints. The constraints imposed by this method are only related to the trajectory length and not to the density of obstacles. However, the classical safe corridors must be computed before trajectory optimization, since they are constructed based on the initial values of the optimization. For motion planning of a tractor-trailer robot, this will take a lot of time, especially if the number of trailers is large, since the safety corridors need to be built for each vehicle of the robot separately. The second disadvantage of the safe corridor is its high dependence on initial value, while a good one requires much more time due to the high state dimension and complex kinematics of the robot. Usually, the process of safe corridor generation selects some state points of a trajectory as seeds, which will affect the location and size of the generated safe corridors. Thus in some scenarios where it is more difficult to obtain a good initial value of the trajectory, e.g., the tractor-trailer robot needs to make a turn in a narrow environment, the feasible region constrained by the safe corridors may not be able to contain sufficiently good solutions. To reduce the dependency of the optimization on initial values, Li et al. [36] proposed a multi-stage method that only uses paths from A* as the initial value for trajectory optimization of tractor-trailer robots traversing a curvy tunnel, improving some of the efficiency for this task.

In this paper, we fully leverage the collision-free regions of the environment, directly applying deformations to trajectories in continuous space, which can avoid the above-mentioned problems brought by safe corridors. In known scenarios, the process for environment can be pre-done to adapt our method, while in unknown environments we can also use separately opened thread for maintenance without occupying the time of trajectory planning.

III. Planning Framework

In this work, we mainly focus on tractor-trailer robots whose tractor has an Ackermann chassis, as shown in Fig. 2. Using the simplified bicycle model [37] and letting the rear center of the tractor of the robot with N trailers is $\boldsymbol{p}_0 = [x_0, y_0]^{\mathrm{T}}$, and the position of each trailer is denoted by $\boldsymbol{p}_i = [x_i, y_i]^{\mathrm{T}}, i = 1, 2, ..., \mathrm{N}$, we can write the kinematic model of the robot as follows:

$$\dot{x}_0(t) = v_0(t)\cos\theta_0(t), \tag{1}$$
$$\dot{y}_0(t) = v_0(t)\sin\theta_0(t), \tag{2}$$
$$\dot{v}_0(t) = a(t), \tag{3}$$
$$\dot{\theta}_0(t) = v_0(t)\frac{\tan\delta(t)}{L_0}, \tag{4}$$
$$\dot{\theta}_1(t) = \frac{v_0(t)\sin(\theta_0(t) - \theta_1(t))}{L_1}, \tag{5}$$
$$\vdots$$
$$\dot{\theta}_i(t) = \frac{v_{i-1}(t)\sin(\theta_{i-1}(t) - \theta_i(t))}{L_i}, \tag{6}$$
$$\vdots$$
$$\dot{\theta}_{\mathrm{N}}(t) = \frac{v_{\mathrm{N}-1}(t)\sin(\theta_{\mathrm{N}-1}(t) - \theta_{\mathrm{N}}(t))}{L_{\mathrm{N}}}, \tag{7}$$

where $t \in (0, +\infty)$ denotes the time, v_0, a, δ, L_0 are the velocity, longitudinal acceleration, steering angle, and wheelbase length of the tractor, respectively. $\theta_i, i = 0, 1, ..., \mathrm{N}$ is the yaw angle of each vehicle in the world frame. For convenience, we use $\boldsymbol{\theta}(t) = [\theta_1(t), \theta_2(t), ..., \theta_{\mathrm{N}}(t)]$ for the yaw angles of all trailers. $v_i = v_{i-1}\cos(\theta_{i-1} - \theta_i), L_i, i = 1, ...\mathrm{N}$ are the velocity and length of the rigid link of each trailer, respectively.

Our goal is to plan a kinematically feasible and collision-free trajectory with the initial state of the robot $[x_0^{\text{init}}, y_0^{\text{init}}, v_0^{\text{init}}, \theta_0^{\text{init}}, \boldsymbol{\theta}^{\text{init}}]^{\mathrm{T}}$ where $\boldsymbol{\theta}^{\text{init}} = [\theta_1^{\text{init}}, \theta_2^{\text{init}}, ..., \theta_{\mathrm{N}}^{\text{init}}]$ and information about the obstacles $\mathbf{Info}_c$, such that the robot can follow this trajectory and eventually arrive within the assigned target region. In this work, point cloud are used to characterize the obstacles information as it can be directly acquired from LiDAR. Besides, we utilize a 2D convex polygon $\mathbf{Info}_e$ to describe the target region, which is a more general condition compared to the work [7] and can also brings convenience for trajectory optimization.

A. Trajectory Parameterization

Differential flatness and polynomial representations have brought a significant improvement in the efficiency of trajectory generation for car-like robots [12]. The trajectory representation in this work is also inspired by these. For the robot like the one in Fig. 2, Deligiannis et al. [38] have demonstrated it is a differential flat system with $\boldsymbol{p}_{\mathrm{N}}$ as flat output. However, the order of the derivatives involved in the derivation process of physical variables such as the longitudinal acceleration and steering angle of the tractor is positively correlated with the number of trailers. Thus the

nonlinearity of the system will sharply strengthen as the number of trailers increases. Meanwhile, this method cannot be extended to more general tractor-trailer robot system, where the trailer $\boldsymbol{p}_{i+1}$ is not hooked directly to the center $\boldsymbol{p}_i$ of the previous one ($i = 1, 2, ..., \mathrm{N}$), but at a distance from this point. [2]

To simplify the robot state representation and to allow the method to be extended to more general tractor-trailer systems, we use $[x_0(t), y_0(t), \boldsymbol{\theta}(t)]^{\mathrm{T}}$ to represent the trajectory of the robot. Combining with Eq.(1)∼Eq.(7), we can compute some useful physical variables as follows:

$$\theta_0(t) = \text{atan2}(\dot{y}_0(t), \dot{x}_0(t)), \tag{8}$$

$$v_0(t) = \sqrt{\dot{x}_0^2(t) + \dot{y}_0^2(t)}, \tag{9}$$

$$a(t) = \frac{\dot{x}_0(t)\ddot{x}_0(t) + \dot{y}_0(t)\ddot{y}_0(t)}{\sqrt{\dot{x}_0^2(t) + \dot{y}_0^2(t)}}, \tag{10}$$

$$\kappa(t) = \frac{\dot{x}_0(t)\ddot{y}_0(t) - \dot{y}_0(t)\ddot{x}_0(t)}{\sqrt{(\dot{x}_0^2(t) + \dot{y}_0^2(t))^3}}, \tag{11}$$

where $\kappa(t)$ is the curvature of the trajectory. However, when the robot is stationary, i.e., there is a moment $t = t_s$, such that $v_0(t_s) = 0$, the above part of the equations will have a zero denominator, which is known as the singularity of differential flatness [13]. To avoid this problem, existing works usually add small positive constants to the denominator [12] or set a minimum velocity constraint [39]. These can lead to larger trajectory tracking errors, as well as making it difficult for the optimizer to converge to a solution that satisfies the kinematic constraints, especially the curvature constraint. To demonstrate this point, we introduce a model predictive control (MPC)-based controller to measure the tracking error. Details can be found in Sec.IV-C and Sec.IV-D.

To solve this problem, inspired by the work [40] that utilizes arc length $s(t)$ to represent trajectories, we introduce *slackened arc length trajectory* $\mathfrak{s}(t)$ and a constraint to avoid singularities. Where s and $\mathfrak{s}$ are both functions of the time t. For brevity, In the rest of the paper, we will simplify them to writing s and $\mathfrak{s}$ instead of $s(t)$ and $\mathfrak{s}(t)$ when necessary. According to the work [40], the arc length $s(t)$ is satisfied: $s(0) = 0, \dot{s}(t) = \sqrt{\dot{x}_0^2(t) + \dot{y}_0^2(t)} = v_0(t) \geq 0$. Also, Eq.(1) and Eq.(2) are rewritten as:

$$\dot{x}_0(s)\dot{s}(t) = \dot{s}(t)\cos\theta_0(t), \tag{12}$$

$$\dot{y}_0(s)\dot{s}(t) = \dot{s}(t)\sin\theta_0(t). \tag{13}$$

Thus, we can rewrite Eq.(8)∼Eq.(11) as follows:

$$\theta_0(t) = \text{atan2}(\dot{y}_0(s)\dot{s}(t), \dot{x}_0(s)\dot{s}(t)), \tag{14}$$

$$v_0(t) = \dot{s}(t), \tag{15}$$

$$a(t) = \ddot{s}(t), \tag{16}$$

$$\kappa(t) = \frac{(\dot{x}_0(s)\ddot{y}_0(s) - \dot{y}_0(s)\ddot{x}_0(s))\dot{s}^3(t)}{\dot{s}^3(t)}. \tag{17}$$

Due to the continuity of the robot's state, for the moment $t = t_s$ with $\dot{s}(t_s) = 0$, we only need to define: $\theta_0(t_s) = \lim_{t\to t_s}\theta_0(t), \kappa(t_s) = \lim_{t\to t_s}\kappa(t)$ and make sure that $\dot{x}_0^2(s(t_s)) + \dot{y}_0^2(s(t_s)) > 0$. At this point, we have solved the singularity problem, but introduced the following constraints:

$$s(t) \geq 0, \tag{18}$$

$$\dot{s}(t) \geq 0, \tag{19}$$

$$\dot{x}_0^2(s) + \dot{y}_0^2(s) = 1 \quad \text{when } \dot{s}(t) \neq 0, \tag{20}$$

$$\dot{x}_0^2(s) + \dot{y}_0^2(s) > 0 \quad \text{when } \dot{s}(t) = 0. \tag{21}$$

For tractor, it is rare for the forward velocity to be zero. Thus, using polynomials to represent the trajectory poses a greater challenge to the optimizer because of the need to impose equation constraint (20) at each timestamp.

To slacken the constraint (20) and facilitate optimization, we define a new variable *slackened arc length* $\mathfrak{s}(t)$ to replace the arc length $s(t)$. In this paper, we follow a part of the definition of the arc length $s(t)$ by setting $\mathfrak{s}(0) = 0$ and $\dot{\mathfrak{s}}(t) \geq 0$. Similarly, we can rewrite Eq.(1) and Eq.(2) as:

$$\dot{x}_{\text{new}}(\mathfrak{s}(t))\dot{\mathfrak{s}}(t) = v_0(t)\cos\theta_0(t), \tag{22}$$

$$\dot{y}_{\text{new}}(\mathfrak{s}(t))\dot{\mathfrak{s}}(t) = v_0(t)\sin\theta_0(t), \tag{23}$$

Where both x_{new} and y_{new} are differentiable functions of $\mathfrak{s}$, $\mathfrak{s}$ is a differentiable function of time t, and $x_{\text{new}}(\mathfrak{s}(t)) = x_0(t), y_{\text{new}}(\mathfrak{s}(t)) = y_0(t)$. Note that Eq.(22) and Eq.(23) can be easily proved to be equivalent to Eq.(1) and Eq.(2), respectively, by the chain rule. Thus, for brevity, we will still use the symbols $x_0(\mathfrak{s}(t))$ and $y_0(\mathfrak{s}(t))$ instead of the symbols $x_{\text{new}}(\mathfrak{s}(t))$ and $y_{\text{new}}(\mathfrak{s}(t))$ in the subsequent parts of the paper.

Subsequently, adding the constraint $\dot{x}_0^2(\mathfrak{s}) + \dot{y}_0^2(\mathfrak{s}) \geq \delta_+$, where $\delta_+ > 0$ is a constant, and using the technique mentioned in the previous paragraph related to the continuity of the robot's state, we can rewrite Eq.(8)∼Eq.(11) as follows:

$$\theta_0(t) = \text{atan2}(\dot{y}_0(\mathfrak{s}), \dot{x}_0(\mathfrak{s})), \tag{24}$$

$$v_0(t) = \dot{\mathfrak{s}}(t)\sqrt{\dot{x}_0^2(\mathfrak{s}) + \dot{y}_0^2(\mathfrak{s})}, \tag{25}$$

$$\begin{aligned} a(t) =& \ddot{\mathfrak{s}}(t)\sqrt{\dot{x}_0^2(\mathfrak{s}) + \dot{y}_0^2(\mathfrak{s})} \\ &+ \dot{\mathfrak{s}}^2(t)\frac{\dot{x}_0(\mathfrak{s})\ddot{x}_0(\mathfrak{s}) + \dot{y}_0(\mathfrak{s})\ddot{y}_0(\mathfrak{s})}{\sqrt{\dot{x}_0^2(\mathfrak{s}) + \dot{y}_0^2(\mathfrak{s})}}, \end{aligned} \tag{26}$$

$$\kappa(t) = \frac{\dot{x}_0(\mathfrak{s})\ddot{y}_0(\mathfrak{s}) - \dot{y}_0(\mathfrak{s})\ddot{x}_0(\mathfrak{s})}{\sqrt{(\dot{x}_0^2(\mathfrak{s}) + \dot{y}_0^2(\mathfrak{s}))^3}}. \tag{27}$$

Same as in the works [12], [41] to consider human comfort by minimizing integration of jerk [42], based on the optimality conditions proved by Wang et al. [43] for the multi-stage control effort minimization problem, we choose quintic piecewise polynomials with four times continuously differentiable at the segmented points as the representation for trajectories of the robot. Introducing $\mathfrak{s}(t)$, each piece of trajectory is denoted as:

$$x_{j0}(\mathfrak{s}) = \boldsymbol{c}_{x_j}^{\mathrm{T}}\gamma(\mathfrak{s}) \qquad \mathfrak{s} \in [0, \mathfrak{s}_j(T_p) - \mathfrak{s}_j(0)], \tag{28}$$

$$y_{j0}(\mathfrak{s}) = \boldsymbol{c}_{y_j}^{\mathrm{T}}\gamma(\mathfrak{s}) \qquad \mathfrak{s} \in [0, \mathfrak{s}_j(T_p) - \mathfrak{s}_j(0)], \tag{29}$$

$$\mathfrak{s}_j(t) = \boldsymbol{c}_{\mathfrak{s}_j}^{\mathrm{T}}\gamma(t) \qquad t \in [0, T_p], \tag{30}$$

$$\theta_{ki}(t) = \boldsymbol{c}_{\theta_{ki}}^{\mathrm{T}}\gamma(t) \qquad t \in [0, T_\theta], \tag{31}$$

where $j = 1, 2, ..., \mathrm{M}; k = 1, 2, ..., \Omega$ is the index of piece-wise polynomial, $T_*, * = \{p, \theta\}$ is the duration of a piece of the trajectory, $c_* \in \mathbb{R}^6, * = \{x_j, y_j, \mathfrak{s}_j, \theta_{ki}\}$ is the coefficient of polynomial, $\gamma(*) = [1, *, *^2, ..., *^5]^{\mathrm{T}}, * = \{t, \mathfrak{s}\}$ is the natural base.

B. Optimization Problem

In this paper, we formulate the trajectory optimization problem for tractor-trailer robots as:

$$\min_{\boldsymbol{C},\boldsymbol{e},T_f} f(\boldsymbol{C},\boldsymbol{e},T_f) = \int_0^{\Sigma_{j=1}^{\mathrm{M}}\mathfrak{s}_j(T_p)} \boldsymbol{j}_p(\mathfrak{s})^{\mathrm{T}}\boldsymbol{W}_p\boldsymbol{j}_p(\mathfrak{s})d\mathfrak{s} \quad (32)$$

$$+ \int_0^{T_f} \boldsymbol{j}_\theta(t)^{\mathrm{T}}\boldsymbol{W}_\theta\boldsymbol{j}_\theta(t)dt + \rho_t T_f \quad (33)$$

$$s.t. \quad \dot{\theta}_i(t)L_i = v_{i-1}(t)\sin(\theta_{i-1}(t) - \theta_i(t)), \quad (34)$$

$$\boldsymbol{M}_p(\boldsymbol{S})\boldsymbol{c}_p = \boldsymbol{b}_p(\boldsymbol{P},\boldsymbol{e}), \quad (35)$$

$$\boldsymbol{M}_{\mathfrak{s}}(T_p)\boldsymbol{c}_{\mathfrak{s}} = \boldsymbol{b}_{\mathfrak{s}}(\boldsymbol{S}), \quad (36)$$

$$\boldsymbol{M}_\theta(T_\theta)\boldsymbol{c}_\theta = \boldsymbol{b}_\theta(\boldsymbol{\Theta},\boldsymbol{e}), \quad (37)$$

$$T_f > 0, \quad \mathfrak{s}(t) \geq 0, \quad \dot{\mathfrak{s}}(t) \geq 0, \quad (38)$$

$$\dot{x}_0^2(\mathfrak{s}) + \dot{y}_0^2(\mathfrak{s}) \geq \delta_+, \quad (39)$$

$$|\theta_{i-1}(t) - \theta_i(t)| \leq \delta\theta_{\max}, \quad (40)$$

$$v_0^2(t) \leq v_{\mathrm{mlon}}^2, \quad a^2(t) \leq a_{\mathrm{mlon}}^2, \quad (41)$$

$$a_{\mathrm{lat}}^2(t) \leq a_{\mathrm{mlat}}^2, \quad \kappa^2(t) \leq \kappa_{\max}^2, \quad (42)$$

$$\boldsymbol{\mathcal{G}}_e(\boldsymbol{e}, \mathbf{Info}_e) \leq \mathbf{0}, \quad (43)$$

$$\boldsymbol{\mathcal{G}}_c(\boldsymbol{C},\boldsymbol{e},T_f, \mathbf{Info}_c) \leq \mathbf{0}, \quad (44)$$

where $\boldsymbol{c}_p = [[\boldsymbol{c}_{x_1}, \boldsymbol{c}_{y_1}]^{\mathrm{T}}, [\boldsymbol{c}_{x_2}, \boldsymbol{c}_{y_2}]^{\mathrm{T}}, ..., [\boldsymbol{c}_{x_{\mathrm{M}}}, \boldsymbol{c}_{y_{\mathrm{M}}}]^{\mathrm{T}}]^{\mathrm{T}} \in \mathbb{R}^{6\mathrm{M}\times 2}$,$\boldsymbol{c}_{\mathfrak{s}} = [\boldsymbol{c}_{\mathfrak{s}_1}^{\mathrm{T}}, \boldsymbol{c}_{\mathfrak{s}_2}^{\mathrm{T}}, ..., \boldsymbol{c}_{\mathfrak{s}_{\mathrm{M}}}^{\mathrm{T}}]^{\mathrm{T}} \in \mathbb{R}^{6\mathrm{M}\times 1}$,$\boldsymbol{c}_\theta = [\boldsymbol{c}_{\theta_1}^{\mathrm{T}}, \boldsymbol{c}_{\theta_2}^{\mathrm{T}}, ..., \boldsymbol{c}_{\theta_\Omega}^{\mathrm{T}}]^{\mathrm{T}} \in \mathbb{R}^{6\Omega\times \mathrm{N}}$, $\boldsymbol{c}_{\theta_k} = [\boldsymbol{c}_{\theta_{11}}, \boldsymbol{c}_{\theta_{12}}, ..., \boldsymbol{c}_{\theta_{1\mathrm{N}}}] \in \mathbb{R}^{6\times \mathrm{N}}$ are coefficient matrices. $\boldsymbol{C} = \{\boldsymbol{c}_p, \boldsymbol{c}_{\mathfrak{s}}, \boldsymbol{c}_\theta\}$. For the other optimization variables, $\boldsymbol{e} = [x_0^{\mathrm{fina}}, y_0^{\mathrm{fina}}, \theta_0^{\mathrm{fina}}, \boldsymbol{\theta}^{\mathrm{fina}}]^{\mathrm{T}}$, $T_f = \mathrm{M}T_p = \Omega T_\theta$ denote the end state of the robot and total duration of the trajectory, respectively, where $\boldsymbol{\theta}^{\mathrm{fina}} = [\theta_1^{\mathrm{fina}}, \theta_2^{\mathrm{fina}}, ..., \theta_{\mathrm{N}}^{\mathrm{fina}}]$.

In the objective function $f(\boldsymbol{C},\boldsymbol{e},T_f)$, $\boldsymbol{j}_p(\mathfrak{s}) = [x_0^{(3)}(\mathfrak{s}), y_0^{(3)}(\mathfrak{s})]^{\mathrm{T}}$ and $\boldsymbol{j}_{\mathfrak{s}\theta}(t) = [\mathfrak{s}^{(3)}(t), \boldsymbol{\theta}^{(3)}(t)]^{\mathrm{T}}$ denote the jerk of the trajectories. Their weighted square integral represents the smoothness of the trajectory. $\boldsymbol{W}_p \in \mathbb{R}^{2\times 2}, \boldsymbol{W}_{\mathfrak{s}\theta} \in \mathbb{R}^{(\mathrm{N}+1)\times(\mathrm{N}+1)}$ are diagonal positive matrices representing the weights. $\rho_t > 0$ is a constant to give the trajectory some aggressiveness.

Eq.(34) denotes the kinematic constraints (6). Eq.(35)∼Eq.(37) denote the combinations of the continuity constraints mentioned in last subsection and boundary conditions of the trajectory:

$$[x_0(0), \dot{x}_0(0)] = [x_0^{\mathrm{init}}, \cos\theta_0^{\mathrm{init}}], \quad (45)$$

$$[y_0(0), \dot{y}_0(0)] = [y_0^{\mathrm{init}}, \sin\theta_0^{\mathrm{init}}], \quad (46)$$

$$[\mathfrak{s}(0), \dot{\mathfrak{s}}(0)] = [0, v_0^{\mathrm{init}}], \quad (47)$$

$$[\theta_i(0), \dot{\theta}_i(0)] = [\theta_i^{\mathrm{init}}, \frac{v_{i-1}^{\mathrm{init}}\sin\theta_{d(i-1)}^{\mathrm{init}}}{L_i}], \quad (48)$$

$$[x_0(\mathfrak{s}_f), \dot{x}_0(\mathfrak{s}_f)] = [x_0^{\mathrm{fina}}, \cos\theta_0^{\mathrm{fina}}], \quad (49)$$

$$[y_0(\mathfrak{s}_f)), \dot{y}_0(\mathfrak{s}_f)] = [y_0^{\mathrm{fina}}, \sin\theta_0^{\mathrm{fina}}], \quad (50)$$

$$[\mathfrak{s}(T_f), \dot{\mathfrak{s}}(T_f)] = [\mathfrak{s}_f, 0], \quad (51)$$

$$[\theta_i(T_f), \dot{\theta}_i(T_f)] = [\theta_i^{\mathrm{fina}}, 0], \quad (52)$$

where $\theta_{d(i-1)}^{\mathrm{init}} = \theta_{i-1}^{\mathrm{init}} - \theta_i^{\mathrm{init}}$, $v_i^{\mathrm{init}} = v_{i-1}^{\mathrm{init}}\cos\theta_{d(i-1)}^{\mathrm{init}}$, $i = 1, 2, ..., \mathrm{N}$. $\boldsymbol{P} \in \mathbb{R}^{2\times(\mathrm{M}-1)}, \boldsymbol{\Theta} \in \mathbb{R}^{\mathrm{N}\times(\Omega-1)}$ are the segment points of the tractor and trailers trajectories, respectively. $\mathfrak{s}_f = \|\boldsymbol{S}\|_1$ and $\boldsymbol{S} \in \mathbb{R}_{>0}^{\mathrm{M}\times 1}$ denote the total length of the *slackened arc length trajectory* and the length of each segment, respectively. Thus $\mathbf{M}_p, \mathbf{M}_{\mathfrak{s}} \in \mathbb{R}^{(6\mathrm{M}-2)\times 6\mathrm{M}}$, $\mathbf{M}_\theta \in \mathbb{R}^{(6\Omega-2)\times 6\Omega}$. In this work, we set $\dot{x}_0^2(0) + \dot{y}_0^2(0) = 1$ and $\delta_+ = 0.9$, which is enough for optimization.

Conditions (38)∼(39) denote the positive duration and the constraints brought by the introduction of $\mathfrak{s}(t)$ mentioned in the last subsection. Conditions (40)∼(42) are dynamic feasibility constraints, including avoiding jackknife [4], limitation of longitudinal velocity $v_0(t)$, longitude acceleration $a(t)$, latitude acceleration $a_{\mathrm{lat}}(t) = v_0^2(t)\kappa(t)$, and curvature $\kappa(t)$, where $\delta\theta_{\max}, v_{\mathrm{mlon}}, a_{\mathrm{mlon}}, a_{\mathrm{mlat}}, \kappa_{\max}$ are constants.

Conditions (43) denote the constraint of the end state of the robot. In this paper, we set the vertices of the 2D convex polygon denoting the target region be $\mathbf{v}_\lambda$, and the normal vectors of corresponding edges toward the outside of the polygon be $\mathbf{n}_\lambda, \lambda = 0, 1, .., \mathrm{N}_e - 1$, as the example of convex pentagon given in the right part of Fig. 2. We only need to ensure that each vertices of each vehicle of the robot are within this region. Let the position of each vertex of each vehicle in the respective body frame be $\boldsymbol{p}_{i\mu}, i = 0, 1, ..., \mathrm{N}, \mu = 1, 2, 3, 4$, we can obtain $4\times(\mathrm{N}+1)\times(\mathrm{N}_e-1)$ constraints: $\mathbf{n}_\lambda^{\mathrm{T}}(\boldsymbol{R}_i^{\mathrm{fina}}\boldsymbol{p}_{i\mu} + \boldsymbol{p}_i^{\mathrm{fina}} - \mathbf{v}_\lambda) \leq 0$, where

$$\boldsymbol{p}_0^{\mathrm{fina}} = [x_0^{\mathrm{fina}}, y_0^{\mathrm{fina}}]^{\mathrm{T}}, \quad (53)$$

$$\boldsymbol{p}_i^{\mathrm{fina}} = \boldsymbol{p}_{i-1}^{\mathrm{fina}} - \boldsymbol{R}_i^{\mathrm{fina}}[L_i, 0]^{\mathrm{T}} \qquad i = 1, 2, ..., \mathrm{N}, \quad (54)$$

$$\boldsymbol{R}_i^{\mathrm{fina}} = \begin{bmatrix} \cos\theta_i^{\mathrm{fina}} & -\sin\theta_i^{\mathrm{fina}} \\ \sin\theta_i^{\mathrm{fina}} & \cos\theta_i^{\mathrm{fina}} \end{bmatrix} \qquad i = 0, 1, ..., \mathrm{N}. \quad (55)$$

In Conditions (43), $\mathbf{0} = [0, 0, ..., 0]^{\mathrm{T}} \in \mathbb{R}^{4(\mathrm{N}+1)(\mathrm{N}_e-1)}$ and the inequality is taken element-wise.

Conditions (44) denote the safety constraints. In this paper, we use 3D point clouds $\mathbf{Info}_c \in \mathbb{R}^{3\times \mathrm{N}_c}$ to represent the environment, further extracting obstacle information from it, where N_c is the number of the points. Subsequently, we construct a Signed Distance Field (SDF) to fuse the information of collision-free region and obstacles. In SDF, the value at each state of the space is the distance to the edge of the nearest obstacle, where the value inside the obstacle is negative, which can be used to directly deform the trajectory of the robot. For example, assume that the vehicles of the

robot can be wrapped by circles of radius r_i and centers $\boldsymbol{p}_{ci}(t), i = 0, 1, ..., \mathrm{N}$, respectively, $\mathscr{S}(\boldsymbol{p}) : \mathbb{R}^2 \mapsto \mathbb{R}$ denotes the value of the SDF, i.e., the signed distance from position $\boldsymbol{p}$ to the edge of the nearest obstacle. We can then impose following $\mathrm{N}+1$ constraints on the optimization problem: $\mathscr{S}(\boldsymbol{p}_{ci}(t)) > r_i, i = 0, 1, ..., \mathrm{N}$, where

$$\boldsymbol{p}_0(t) = [x_0(\mathfrak{s}(t)), y_0(\mathfrak{s}(t))]^{\mathrm{T}}, \tag{56}$$

$$\boldsymbol{p}_{c0}(t) = \boldsymbol{p}_0(t) + [L_{\text{rear}} \cos\theta_0(t), L_{\text{rear}} \sin\theta_0(t)]^{\mathrm{T}}, \tag{57}$$

$$\boldsymbol{p}_{ci}(t) = \boldsymbol{p}_i(t) = \boldsymbol{p}_{i-1}(t) - \boldsymbol{R}_i(t)[L_i, 0]^{\mathrm{T}}, \tag{58}$$

$$\boldsymbol{R}_i(t) = \begin{bmatrix} \cos\theta_i(t) & -\sin\theta_i(t) \\ \sin\theta_i(t) & \cos\theta_i(t) \end{bmatrix}, \qquad i = 1, ..., \mathrm{N}. \tag{59}$$

L_{rear} is the distance between $\boldsymbol{p}_0$ and the center of the tractor. The inequality in Conditions (44) is also taken element-wise. Conditions (44) are also include vehicle-to-vehicle collision avoidance. We impose the constraint that the centers $\boldsymbol{p}_{ci}, i = 0, 1, ..., \mathrm{N}$ of each vehicle of the robot must be greater than a threshold distance from each other.

C. Problem Simplification and Solving

Observing Eq.(35)∼Eq.(37), we can see that the value of $\boldsymbol{C}$ is related to $\boldsymbol{P}, \boldsymbol{S}, \boldsymbol{\Theta}, \boldsymbol{e}, T_f$. Inspired by the work [43], we add the constraint of the trajectories having zero second-order derivatives at the start and termination time to ensure that the matrices $\boldsymbol{M}_p, \boldsymbol{M}_{\mathfrak{s}}, \boldsymbol{M}_\theta$ are invertible. Thus using the method proposed in the work [43] to convert the optimization variables $\{\boldsymbol{C}, \boldsymbol{e}, T_f\}$ to $\{\boldsymbol{P}, \boldsymbol{S}, \boldsymbol{\Theta}, \boldsymbol{e}, T_f\}$, we can eliminate the equation constraints (35)∼(37) while reducing the dimension of the optimization variables.

To deal with the constraints $T_f > 0, \mathfrak{s}(t) \geq 0$, we refer to the differential homogeneous mapping [44], using the following computationally convenient C^2 function to convert each element of $\boldsymbol{S} \in \mathbb{R}^{\mathrm{M}\times 1}_{>0}$ and $T_f \in \mathbb{R}_{>0}$ to $\mathfrak{S} \in \mathbb{R}^{\mathrm{M}\times 1}$ and $\tau_f \in \mathbb{R}$, respectively:

$$\mathrm{L}_{c2}(x) = \begin{cases} 1 - \sqrt{2x^{-1} - 1} & 0 < x \leq 1 \\ \sqrt{2x - 1} - 1 & x > 1 \end{cases}. \tag{60}$$

With the constraint $\dot{\mathfrak{s}}(t) \geq 0$, we can then ensure $\mathfrak{s}(t) \geq 0$. Also, to guarantee that constraint (40) is satisfied when $t = T_f$, we use an inverse sigmoid-like function to convert $\boldsymbol{\theta}^{\text{fina}}$ to $\boldsymbol{\vartheta}_d^{\text{fina}} = [\vartheta_{d1}^{\text{fina}}, \vartheta_{d2}^{\text{fina}}, ..., \vartheta_{d\mathrm{N}}^{\text{fina}}]$,the corresponding equations are as follows:

$$\theta_{di}^{\text{fina}} = \theta_{i-1}^{\text{fina}} - \theta_i^{\text{fina}} \qquad i = 1, 2, ..., \mathrm{N}, \tag{61}$$

$$\vartheta_{di}^{\text{fina}} = \mathrm{L}_{c2}\left(\frac{\delta\theta_{\max} + \theta_{di}^{\text{fina}}}{\delta\theta_{\max} - \theta_{di}^{\text{fina}}}\right) \qquad i = 1, 2, ..., N. \tag{62}$$

For the remaining kinematic constraints (34) and other inequality constraints, we discretize each piece of the duration T_p as K time stamps $\tilde{t}_p = (p/\mathrm{K}) \cdot T_p, p = 0, 1, ..., K-1$, and impose the constraints on these time stamps.

Then, we use Powell-Hestenes-Rockafellar Augmented Lagrangian method [45] (PHR-ALM) to solve the simplified problem, which is an effective method for solving large-scale problems and those with complex constraints.

Inspired by other works on robot trajectory optimization problems that also need to deal with equation constraints [41], [46], we set the trajectory of trailers have more pieces (i.e. $\Omega > \mathrm{M}$) to make it easier for the optimizer to converge and better fit the kinematic constraints (34). Based on engineering experience, we found that the solver can achieve good efficiency and success when $\Omega = \text{ceil}(2.0 \times \mathrm{M})$, where function $\text{ceil}(x)$ serves to take the smallest integer not less than x.

Algorithm 1: SE(2)-MHA: Fast multi-terminal path searching based on Hybrid-A* for tractor-trailers

Input: grid map $\mathcal{M}$, start state $\boldsymbol{s}_{\text{start}} \in \mathbb{R} \times SO(2)^{\mathrm{N}+1}$, target region $\textbf{Info}_e$, $d_{\text{shoot}} > 0$, weights $\boldsymbol{w}_g, w_l, w_e$.
Output: the best path $\mathcal{P}$
begin
  $\mathcal{O}, \mathcal{C}, \mathcal{E} \leftarrow \varnothing, \mathcal{T} \leftarrow$ GetEnds($\textbf{Info}_e$), $l = +\infty$;
  $n_s \leftarrow \mathcal{O}$.AddStart($\boldsymbol{s}_{\text{start}}$), $\mathcal{E}$.Insert(n_s);
  while $\neg\ \mathcal{O}$.Empty() **do**
    $n_c \leftarrow \mathcal{O}$.Pop(), $\mathcal{C}$.Insert(n_c), $\mathcal{E}$.Insert(n_c);
    for ***each*** $\tau \in \mathcal{T}$ **do**
      if HasPath(τ) **then**
        continue;
      if ReachEnd(n_c, τ) $\vee$ (Dist(n_c, τ) < d_{shoot} $\wedge$ ShootEnd(n_c, τ)) **then**
        $\mathcal{P}_a \leftarrow$ GetFullPath(n_c);
        $l_a =$ CalcCost($\mathcal{P}_a, \textbf{Info}_e, w_l, w_e$);
        if $l_a < l$ **then**
          $l = l_a, \mathcal{P} = \mathcal{P}_a$;
    if AllPathFound() **then**
      return $\mathcal{P}$.
    for ***each*** $c \in$ GetInputs() **do**
      $\boldsymbol{s}_c \leftarrow$ StateTrans(n_c, c);
      if $\neg$ InMap($\boldsymbol{s}_c, \mathcal{M}$) **then**
        continue;
      $n_t \leftarrow \mathcal{E}$.Find($\boldsymbol{s}_c$);
      if (n_t ***is*** Null $\vee\ \neg\ \mathcal{C}$.Find(n_t))$\wedge$ NoCollision(n_c, c) **then**
        $g_t \leftarrow n_c.g +$ GetCost($n_c, c, \boldsymbol{w}_g$);
        if $\neg\ \mathcal{O}$.Find(n_t) **then**
          $\mathcal{O}$.SetAndInsert(n_t), $\mathcal{E}$.Insert(n_t);
        else if $g_t > n_t.g$ **then**
          $n_t.g \leftarrow g_t, n_t.parent \leftarrow n_c$;
          $n_t.f \leftarrow g_t +$ GetHeu($n_t, \textbf{Info}_e$);
  return $\varnothing$.

D. Initial Value Acquisition

There are many methods that can be used to obtain initial values for trajectory optimization. Work [36] uses the A* algorithm search directly for all vehicles of the robot to obtain the path quickly, but since this path is kinematically infeasible, the optimization will take longer to converge to

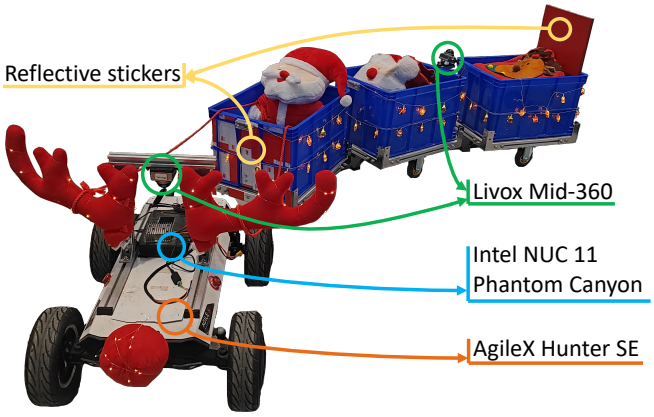


Fig. 3: Tractor-trailer robot. It consists of three DIY trailers and an AgileX Hunter SE as the tractor.

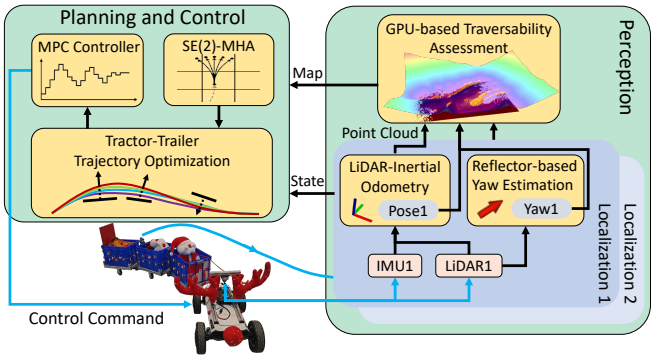


Fig. 4: Software system of the robot. The perception module includes the global localization of the robot, its own state estimation, and traversability assessment. The planning and control module includes the proposed planning pipeline and trajectory tracking controller.

a good solution. Work [3] proposed an extended Hybrid-A* algorithm [14] that searches the trajectory in the space $\mathbb{R} \times SO(2)^{\mathrm{N}+1}$ including the position of the tractor and yaw angles of all vehicles, which can obtain a better initial value. However, due to the large search space, it usually takes much more time, which leads to the whole system not meeting the requirement of real-time replanning in unknown environments for avoiding just-appearing obstacles.

To balance the quality and efficiency, we propose a **m**ulti-terminal path search method based on **H**ybrid-**A*** [14] SE(2)-MHA, which searches only in $SE(2)$ space for the tractor and uses an approximate integration method similar to that in work [3] to obtain the path and terminal position of the trailers in searched paths corresponding to each terminal. Finally, we evaluate the quality of the paths using the weighted cost of the path length and the distance of each vehicle from the target region, choosing the one with the smallest value as the initial value for trajectory optimization. Specifically, assuming the length of the tractor path is l_{tra} and the distance between the terminal position of i-th trailer and the target convex area is $d_{it}, i = 1, 2, \mathrm{N}$, the cost of the path is $\mathrm{Cost}_{traj} = w_{\mathrm{r2}} l_{\mathrm{tra}} + w_{\mathrm{trailer}} \sum_{i=1}^{\mathrm{N}} d_{it}$, where w_{r2} and w_{trailer} are the weights. In our algorithm, d_{it} will be set to zero if the terminal position of the i-th trailer is within the target convex area. The pseudo code of the proposed method is given in Algorithm 1.

Fig. 5: Part of the indoor experimental area. Tractor-trailer robot is traversing complex indoor environment.

In Algorithm 1, $\mathcal{O}$ and $\mathcal{C}$ refer to the open and closed set respectively. While $\mathcal{E}$ denotes is used to store the expanded nodes for pruning to speed up the process of searching. $\mathcal{T}$ includes different end states, each state $\{\boldsymbol{p}_\lambda^e, \theta_\lambda^e\}$ is determined by the midpoint $\boldsymbol{p}_{\lambda c}$ of each edge of the convex polygon and the length L_{f} of the tractor, $\boldsymbol{p}_\lambda^e = \boldsymbol{p}_{\lambda c} - (L_{\mathrm{rear}} + 0.5L_{\mathrm{f}})\mathbf{n}_\lambda, \theta_\lambda^e = \mathrm{atan2}(\mathbf{n}_\lambda(1), \mathbf{n}_\lambda(0))$.To further accelerate the process, we also adopt Dubins Curve [47] to shoot the end states for earlier termination when the distance between the current node and the end state is less than a threshold d_{shoot}. Once we get a feasible path for the tractor, v_0 and the change in θ_0 between two neighboring states in the path will be estimated. We then approximate the states of the trailers by discretizing Eq.(6). The state transfer equations Eq.(1),(2) and (4) are also be discretized for expanding the state with the discretized v_0 and δ as control inputs. In this work, we use a weighted summation of different values as the cost function g, including the distance of expansion, the magnitude of the change in yaw angle θ_0, and the accumulation of control efforts with respect to time. For the heuristic function h, we chose the shortest Euclidean distance to the center of the target region.

IV. Results

A. Implementation details

To validate the performance of our method in real-world applications, we deploy it on a tractor-trailer robot, as shown in Fig. 3. The robot is driven by an Ackermann chassis and carries three easily removable passive trailers. We use hinges that connect the trailer to the tractor and the trailers to the trailers.

Fig. 4 illustrates the software system of the robot. In the perception module, we use the combination of LiDARs and reflective stickers [48], [49] to acquire the state of the robot, which includes the pose of the tractor and the yaw angle of each trailer. Specifically, the reflective stickers of the latter vehicle provides a high-intensity point cloud for the former LiDAR-installed vehicle. Then the relative pose of the two vehicles can be obtained by filtering this part of the point

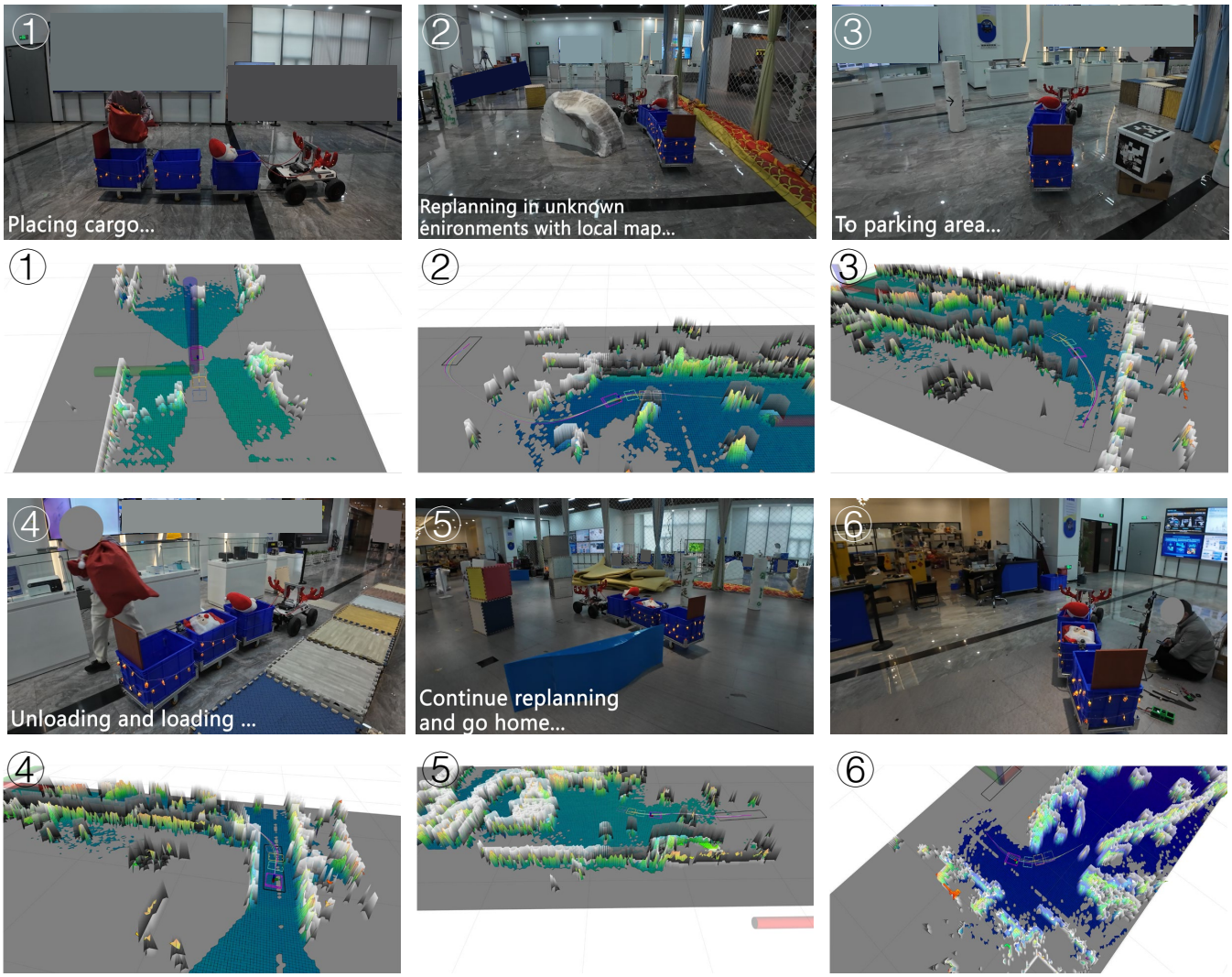


Fig. 6: The whole process of the indoor experiment. Different numerical markers indicate third-person viewpoints or corresponding views from Rviz tacken at different moments. In Rviz, the coloured part indicates the robocentric elevation map, and the semi-transparent black and white part indicates the traversability map, with higher indicating less traversable.

cloud and computing the normal vector of the fitted plane. We run LiDAR-inertia Odometry (LIO) on two LiDARs separately. Using the initial point cloud map constructed by the LiDAR on the tractor as initial map of the LIO on the trailer, we can align the coordinate system of these two LIOs. In this work, we adopt FAST-LIO2 [50] as the LIO algorithm.

The registered point clouds from the tractor and states of the robot are further used to construct elevation map [51] and perform traversability assessment. For real-time requirements, we remove dynamic obstacles using ray casting and compute the covariance matrix of the point cloud in parallel with the help of GPU [52]. Further, still in parallel, we patch unseen grids using the nearest-neighbour method, computing the normal direction of the fitted plane and the approximate curvature [53] for each grid. We obtain the final robocentric occupancy grid map by setting thresholds for traversability. Subsequently, the resulting map is delivered to the planning and control module, where the SDF will be computed by an efficient $O(n)$ algorithm [54].

A replanning process is activated when a set timer is triggered or the target region is updated. It will call the SE(2)-MHA algorithm with the current or predicted state of the robot as the starting state and perform trajectory optimization. A MPC controller will receive the optimized trajectory to compute control commands consisting of speed and steering angle, which utilizes the position of the tractor and the yaw angles of all vehicles of the robot as state. The objective function includes the norm of the error between the states and the reference states, as well as the norm of the control inputs. We use Casadi [55] to construct the MPC problem, solving it by Sequential Quadratic Programming.

We also build some simulation environments based on ROS[1] to testify the effectiveness of our method in various scenarios and conduct comparative experiments with other algorithms, as detailed in Sec.IV-C and Sec.IV-D. All simu-

[1]https://www.ros.org/

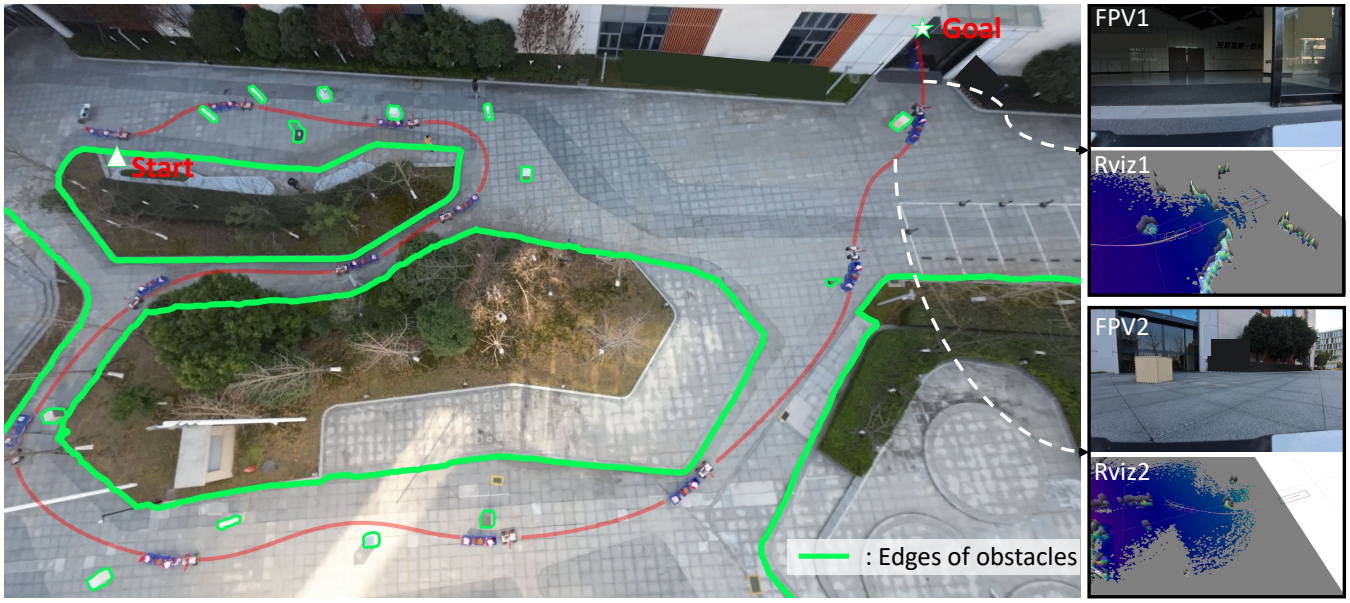


Fig. 7: The second scene of outdoor experiments. The tractor-trailer robot autonomously traverses the park using the proposed planning algorithm to enter the interior of the building through an automatic door.

TABLE I: Statistics in Real-world Experiments

Statistics	Exp.	Mean	Max	STD.
$\hat{v}_0$ (m/s)	Indoor	1.03	1.20	0.215
	Outdoor (1^{st})	1.05	1.20	0.188
	Outdoor (2^{nd})	1.07	1.20	0.184
$\hat{a}$ (m/s^2)	Indoor	0.0651	0.649	0.0685
	Outdoor (1^{st})	0.0758	0.625	0.0758
	Outdoor (2^{nd})	0.0878	0.739	0.0883
$\hat{a}_{\text{lat}}$ (m/s^2)	Indoor	0.117	0.754	0.102
	Outdoor (1^{st})	0.139	0.641	0.111
	Outdoor (2^{nd})	0.157	1.16	0.125
$\hat{\delta\theta}$ (rad)	Indoor	0.137	0.427	0.0930
	Outdoor (1^{st})	0.110	0.442	0.0887
	Outdoor (2^{nd})	0.0899	0.452	0.0828
Tracking Error (m)	Indoor	0.115	0.172	0.0226
	Outdoor (1^{st})	0.115	0.248	0.0270
	Outdoor (2^{nd})	0.114	0.242	0.0211

lations are run on Ubuntu 20.04 with an Intel i9-14900 CPU and a GeForce RTX 4090D GPU.

B. Real-World Experiments

We present several experiments in both indoor and outdoor cluttered environments. In the indoor experiment, we set up the tractor-trailer robot to transport, load and unload cargo to demonstrate the possible real-life applications of the proposed navigation system, as illustrated in the screenshots of Fig. 6. At the beginning, the cargo is placed into one trailer. Receiving the target loading and unloading area, the robot starts to continuously replan the trajectory using the real-time updated map to avoid unknown obstacles. After traversing the complex environment, as shown in Fig. 5, and arriving at the parking area where its cargo is loaded and unloaded, the robot continues to replan and returns to the starting area.

To validate the effectiveness of proposed framework in more scenarios, we conduct two outdoor experiments. The first outdoor scene is the aisle by the non-motorized parking area, as shown in Fig. 1. This environment is filled with different kinds of obstacles including cube-shaped, cylindrical, carts, cars, bicycles, e-bikes etc. The robot needs to constantly replan to avoid obstacles and traverse this narrow passages. The second outdoor experiment is illustrated in Fig. 7. In this scenario, the robot is required to load some cargo and drive into a building by approaching predetermined waypoints. Similar to indoor experiments, the robot perceives the world through the local traversability map updated in real time, and continuously replans to avoid unknown obstacles along the way. Eventually, it enters the building by triggering an automatic door.

In the real-world experiments, limitation of longitudinal velocity, longitude acceleration, latitude acceleration, curvature and difference in yaw angle between adjacent vehicles are set to $v_{\text{mlon}} = 1.2m/s$, $a_{\text{mlon}} = 0.8m/s^2$, $a_{\text{mlat}} = 1.2m/s^2$, $\kappa_{\text{max}} = 0.6m^{-1}$ and $\delta\theta_{\text{max}} = 0.9$ rad, respectively. Meanwhile, we set the time weight $\rho_t = 10.0$ to ensure the aggressiveness of the trajectory.

Tab. I shows the dynamic evaluation metrics during the real-world experiments for the robot, where $\hat{v}_0$ is the estimated longitudinal velocity of the tractor, given by the wheel tachometer. $\hat{a}$ and $\hat{a}_{\text{lat}}$ are measured longitude acceleration and latitude acceleration, respectively, obtained from the

TABLE II: Key Parameter Settings for Simulation in Parking Lot. M.P. represents the proposed method. M.O. represents OCP-STC [7]. The specific names of abbreviations m.s., mu.s., b.p., l.s., m.c.t are "mem size", "mu strategy", "bound push", "linear solver", "max cpu time", respectively.

Type	Para.	Description	Setting
M.P.	L_q	Intervals for taking state points from the front-end path as optimization variables	5 m
	K	The number of points sampled to impose constraints in each polynomial trajectory	16
	m.s.	The number of corrections to approximate the inverse hessian matrix in L-BFGS	64
	δ_{inner}	Desired convergence tolerance for L-BFGS	1.0^{-4}
	past	Distance for delta-based convergence test	3
	N_{iter}	Maximum iteration number of L-BGFS	10000
	ϵ_{cond}	Desired convergence tolerance for ALM	1.0^{-1}
	N_{out}	Maximum iteration number of ALM	100
M.O.	L_{fe}	Intervals for initializing finite elements from the front-end path	0.3 m
	mu.s.	Update strategy for barrier parameter	adaptive
	b.p.	Desired minimum absolute distance from the initial point to bound	$1.0e^{-8}$
	l.s.	The core linear solver for Ipopt	MA97
	m.c.t	Limit on CPU seconds that Ipopt can use to solve one problem	200.0 s

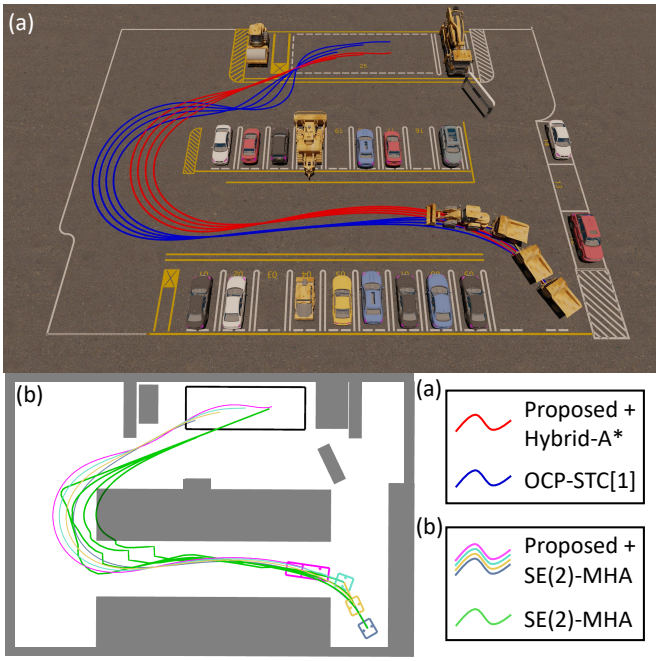


Fig. 8: A bulldozer pulls three transport trucks into a parking space in a parking lot. Figure (a) visualizes this task and the trajectories generated using Hybrid-A* as front end. In this case, the distribution of the obstacles is shown by the gray area in Figure (b).

IMU accompanying the LiDAR. $\hat{\delta\theta}$ is the largest estimated difference between the yaw angles of two neighboring vehicles, read from the estimated robot state. Tracking Error is the distance between the current position and the reference position of the tractor. By integrating $\hat{v}_0$ with respect to time, we calculate the total distance traveled by the tractor to be $36.178m$, $26.774m$ and $81.987m$ in the indoor, first and second outdoor experiments, respectively.

As we can see from Tab. I, the robot maintains a high speed and small tracking error when traversing the environment. The small mean and variance of the latitude and longitude accelerations indicate that the robot can keep a high stability during driving despite the need for constantly replanning when facing unknown environments. In addition, the maximum value of each metric reveals that there is no violation of the constraints we set for the robot. More demonstrations can be found in the attached multimedia.

C. Cases Study

In simulations, we first perform a qualitative and simple quantitative comparison for proposed method and OCP-STC [7] in a parking lot scenario with two different initial value acquisition methods, as shown in Fig. 8, where OCP-STC [7] is an efficient trajectory optimization method for tractor-trailer robots in unstructured environments, which uses Ipopt [56] as the solver and imposes obstacle avoidance constraints using safe travel corridors. In this case, we set $L_0 = 2.7m, \mathrm{N} = 3, L_1 = L_2 = L_3 = 4.0m$. The width of the tractor and trailers $L_w = 2.0m$. The constants related to kinematic constraints are set to $v_{\text{mlon}} = 10.0m/s, a_{\text{mlon}} = 5.0m/s^2, a_{\text{mlat}} = 10m/s^2, \kappa_{\text{max}} = 0.95m^{-1}, \delta\theta_{\text{max}} = 1.47\text{rad}$, respectively. As in the original paper [7], we use AMPL [57] to construct the optimization problem for OCP-STC [7]. Moreover, the list of key parameter settings for optimization algorithms is given in Table II. The results using modified Hybrid-A* proposed in the work [3] as the initial value are shown in Fig. 8(a). Since the presence of narrow spaces and large turns in this scenario, it is difficult to provide good seeds from the front-end path for corridor generation for every vehicle in the robot, the OCP-STC [7] causes the robot to take bigger turns and does not make good use of the collision-free area at the edge of the parking space.

The results of the quantitative comparison are demonstrated in Table III. Since the SE(2)-MHA expands the states only in $SE(2)$ space for the tractor and does not perform collision check on the trailers, the time consumed by path search is reduced by almost an order of magnitude compared to Hybrid-A*. Although such an initial value leads to a longer optimization time, it reduces the total planning time. The corridor-based approach OCP-STC [7] requires completely collision-free initial values for corridor generation. Thus SE(2)-MHA cannot be used as an initial value. In addition, the corridors narrow the solution space of the problem, resulting in it not containing a better solution in this case, giving a longer trajectory length.

The second case is an ablation experiment to demonstrate the importance of *slackened arc length* in a simulated village, as shown in Fig. 9, where v_s is the mini-

TABLE III: Statistics in a parking lot

Front End Method	Search Time (ms)	Optimization Method	Corridor Generation Time (ms)	Optimization Time (ms)	Planning Total Time (s)	Trajectory Length (m)
Hybrid-A*	1224	OCP-STC	27.7	2892.5	4.14	100.7
		Proposed	0.0	156.36	1.38	86.75
SE(2)-MHA	178.7	OCP-STC	/	/	/	/
		Proposed	0.0	185.53	**0.364**	**86.66**

TABLE IV: Tracking error statistics for ablation experiment.

Method	Mean T.E. (m)	Max T.E. (m)	STD. (m)
Proposed w.o. $\mathfrak{s}$ $v_s = 0.01$	0.227	0.434	0.0959
Proposed w.o. $\mathfrak{s}$ $v_s = 0.1$	0.191	0.366	0.0868
Proposed w.o. $\mathfrak{s}$ $v_s = 1.0$	0.443	1.06	0.259
Proposed w. $\mathfrak{s}$	**0.133**	**0.254**	**0.0595**

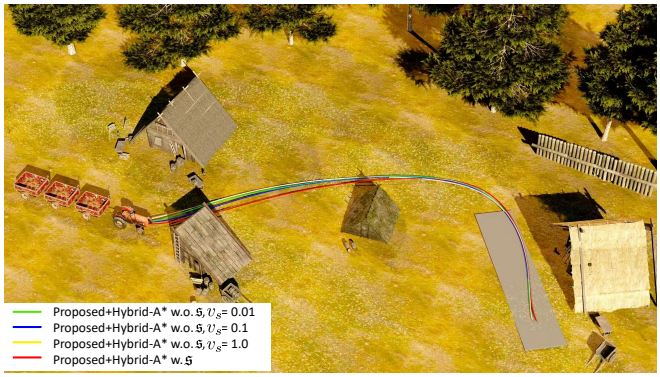
mum speed constrained to avoid singularities. In this case, we set $L_0 = 2.6m, \mathrm{N} = 3, L_1 = L_2 = L_3 = 4.0m$. The width of the tractor and trailers $L_w = 2.8m$. The constants related to kinematic constraints are set to $v_{\text{mlon}} = 10.0m/s, a_{\text{mlon}} = 5.0m/s^2, a_{\text{mlat}} = 2m/s^2, \kappa_{\text{max}} = 0.99m^{-1}, \delta\theta_{\text{max}} = 1.47$rad, respectively. Moreover, the key parameters of the optimization algorithm are set the same as those given in Table II for methods with and without the introduction of slackened arc length $\mathfrak{s}$. The black dashed lines in Fig. 9(b) indicate the boundaries of the constraints. As can be seen in Fig. 9(b), the smaller the v_s, the more significant the effect of singularity on trajectory optimization. Although the smaller the v_s, the more the optimizer is able to constrain the maximum curvature of the robot, it will make the speed of the tractor mutation at the initial and end states more severe, which is not in accordance with the real-world situation, and may cause difficulties for accurate trajectory tracking. We introduce a model predictive control (MPC)-based controller [58] to measure the tracking error in the simulation. In our paper, the tracking error of a tractor-trailer robot is defined as the average tracking error of all vehicles in the robot. The statistical results are shown in Table IV and the curves in Fig. 9(b). Optimized trajectories without the use of $\mathfrak{s}$ lead to larger tracking errors due to abrupt changes in velocity or constraints violations during the initial and end phases of the trajectories. Especially when $v_s = 1.0$, extremely discontinuous velocities cause the controller to almost diverge at the beginning.

D. Benchmark Comparisons

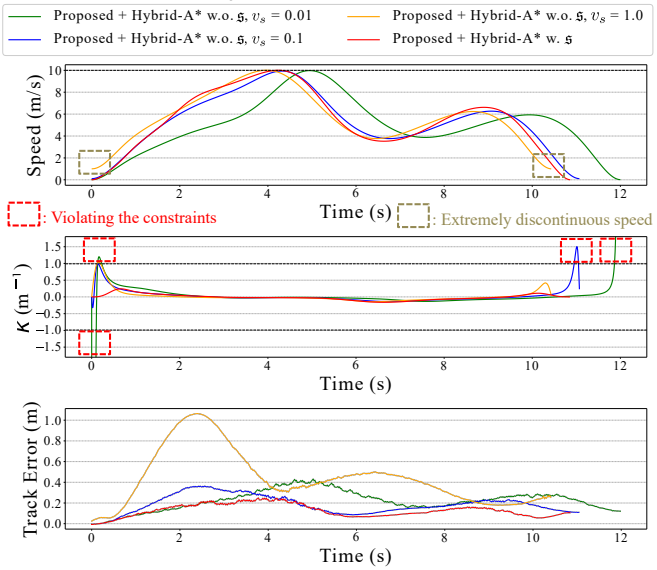
We carry out benchmark comparisons in a simulation environment of size $40m \times 40m$ with random triangular, quadrilateral, and pentagonal obstacles as shown in Fig. 10. We randomly initialize the initial state and target region of the robot in this environment. Among them, the initial state of the robot includes the $SE(2)$ state of the tractor and the yaw angle of each trailer. We use a rectangle as the target region, given that parking areas in the real world are usually of this shape. We set the length and width of the tractor trailer be $0.6m$ and $0.4m$, respectively. The length and width of the trailer are both set to be $0.4m$, the maximum speed is $2.0m/s$, the maximum longitudinal acceleration is $2.0m/s^2$, the minimum angle at which jacknife occurs is 1.47rad. The wheelbase length of the tractor is set to $0.5m$, and the maximum steering angle of it is 0.7 rad. The parameters of all algorithms are finely tuned for the best performance. Moreover, the key parameters for optimization algorithms are consistent with those in Table II, except that considering the scene and robot size, we set $L_q = 1.0$ m, $K = 8$, and $L_{fe} = 0.1m$. As in Sec.IV-C, we follow the original paper [7] and deploy OCP-STC using AMPL [57].

Two initial value acquisition methods and three optimization methods are tested in the above-mentioned environments. The initial value acquisition methods are the proposed SE(2)-MHA and Hybrid-A*, where the latter is the path search method proposed in the work [3], which does exploration over the entire $\mathbb{R} \times SO(2)^{\mathrm{N}+1}$ state space. The optimization methods include OCP-STC [7], Proposed w.o. $\mathfrak{s}$ and Proposed w. $\mathfrak{s}$, where w.o. $\mathfrak{s}$ or w. $\mathfrak{s}$ is with or without using the *slackened arc length* as optimization variables. We compare these method in different environments with different density of obstacles when number of trailers N is 1, 2, or 3. Besides, tasks with different distance from the center of the rear wheels of the tractor to the center point of the target area ($3m \sim 10m$, $10m \sim 20m$, $20m \sim 40m$) are conducted separately. More than one hundred comparison tests are performed in each case.

Fig. 11 illustrates how the trajectory optimization time varies with the problem dimensions. Regardless of which method is used to obtain the initial values, Proposed with *slackened arc length* $\mathfrak{s}$ consumes the least amount of time in optimization. As the number of trailers increases and the target region becomes farther away, this method also shows a slower increase in computational time consumption. This is due to the fact that for the similar problem dimension, compared to OCP-STC [7], we utilize polynomial trajectories and differential flatness to reduces the dimensionality of the optimization variables. Meanwhile, despite the fact that we introduce *slackened arc length* and increase optimization



(a) Trajectories in the simulation.



(b) Speed, curvature, tracking error curves of optimized trajectories.

Fig. 9: The tractor pulls three transport trailers through the simulated village environment. The curves in Figure (a) represent the trajectories of the tractor.

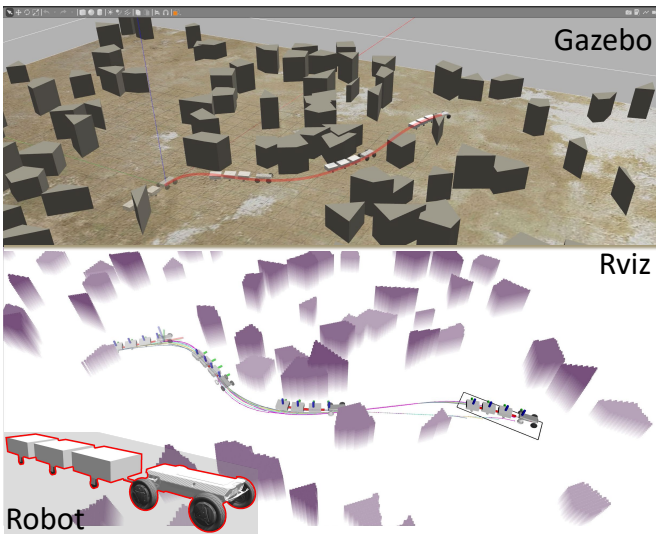


Fig. 10: Physical simulation based on Gazebo. In Rviz, the purple parts indicate obstacles. The height of the point cloud from low to high is indicated by the color from white to purple.

variables, the constructed trajectory optimization problem has no numerical problems caused by singularities, which in turn makes the optimization easier to converge to the local optimum and thus shortens the optimization time. Without *slackened arc length*, the effects from singularities are more and more exposed as the problem becomes more complex and more constraints are imposed. Thus the optimization efficiency decreases drastically. In addition, it can also be seen from Fig. 11 that the dependency of OCP-STC [7] on the initial value is strong. The collision-free path provided by Hybrid-A* makes it possible to reduce the optimization time significantly.

Fig. 12 shows the effect of different initial value selection methods on the success rate of the three different methods when the obstacle density is medium. Where OCP-STC [7] is considered to fail after the optimization consumes more than 30 seconds. Proposed w.o./w. $\mathfrak{s}$ is considered to fail when the iterations of PHR-ALM is greater than 50. Since Hybrid-A* searches for paths in the full state and can provide collision-free initial values, the optimizers all have a higher success rate compared to using SE(2)-MHA. However, the need to generate corridors for all vehicles based on the initial values may lead to problems infeasible in some narrower environments, which lead to higher failure rates when optimization method is OCP-STC [7]. Without $\mathfrak{s}$, singularities lead to poor initial values of the optimization problem, resulting in the optimizer may not converge in a finite number of steps or there is a probability of optimizing to a local minima that cannot satisfy the constraints. Thus there is also a higher failure rate when using Proposed w.o. $\mathfrak{s}$. In the case of using SE(2)-MHA, it may not be possible to generate corridors by initial value as in Fig. 8, leading to a significantly higher failure rate of using OCP-STC [7]. The poorer initial values also amplify the problem of singularities to Proposed w.o. $\mathfrak{s}$ even more, leading to higher failure rates. SE(2)-MHA also leads to more failures in Proposed w. $\mathfrak{s}$, which is mainly attributed to the incompleteness of SE(2)-MHA. It sacrifices the feature of collision-free path for sake of efficiency and thus has a higher probability of generating initial values that make it impossible for the optimizer to optimize to a feasible local minima.

More results on efficiency and quality are summarized in Table V. Where the tuple (x, y, z) under environment complexity denotes the number of triangular, quadrilateral, and pentagonal random obstacles respectively. l_{traj} denotes the mean length of the tractor trajectory, t_{d} denotes the mean time duration of the optimized trajectory, and κ denotes the mean curvature of the tractor, reflecting the smoothness of the trajectory. We set the front-end path search of greater than 5 seconds to count as failure. From the table, it can be seen that the success rate and computation time of SE(2)-MHA do not change much as the number of trailers becomes more. Its success rates are all greater than 98% and time consumption

TABLE V: Benchmark Comparison in Different Cases

N	Environments		Low-Complexity (10, 10, 10)			Medium-Complexity (20, 20, 20)			High-Complexity (40, 40, 40)		
	Method		l_{traj} (m)	t_{d} (s)	$\kappa(\text{m}^{-1})$	l_{traj} (m)	t_{d} (s)	$\kappa(\text{m}^{-1})$	l_{traj} (m)	t_{d} (s)	$\kappa(\text{m}^{-1})$
1	Hybrid-A*	OCP-STC	17.80	11.68	0.3267	17.98	12.83	0.3381	18.68	15.08	0.3593
	(89.57%)	Proposed w.o. $\mathfrak{s}$	17.46	10.58	0.3829	17.38	10.55	0.4198	17.55	10.60	0.4163
	(926.2ms)	Proposed w. $\mathfrak{s}$	**17.27**	**9.959**	**0.2489**	**17.23**	**9.940**	**0.2595**	**17.39**	**10.00**	**0.2603**
	SE(2)-MHA	OCP-STC	17.69	11.24	0.3389	18.44	13.75	0.3458	19.32	20.04	0.5986
	(98.49%)	Proposed w.o. $\mathfrak{s}$	17.32	10.61	0.4935	17.29	10.56	0.5095	17.84	11.11	0.5504
	(115.7ms)	Proposed w. $\mathfrak{s}$	17.30	10.05	0.2469	**17.23**	9.956	0.2474	17.72	10.24	0.2733
2	Hybrid-A*	OCP-STC	18.14	11.41	0.2687	18.76	13.65	0.2955	18.87	17.79	0.2948
	(84.80%)	Proposed w.o. $\mathfrak{s}$	17.88	10.80	0.3629	18.11	11.09	0.3609	17.75	10.81	0.3735
	(904.6ms)	Proposed w. $\mathfrak{s}$	**17.58**	**10.10**	**0.2353**	**17.84**	**10.23**	**0.2395**	**17.53**	**10.07**	**0.2416**
	SE(2)-MHA	OCP-STC	18.57	12.18	0.3017	19.92	18.11	0.3171	20.49	29.39	0.3401
	(98.53%)	Proposed w.o. $\mathfrak{s}$	18.02	11.27	0.5010	18.29	11.19	0.4965	18.75	11.44	0.4731
	(132.1ms)	Proposed w. $\mathfrak{s}$	17.69	10.17	0.2470	18.04	10.34	0.2410	18.49	10.56	0.2535
3	Hybrid-A*	OCP-STC	19.30	12.72	0.2799	19.10	13.90	0.2710	19.44	18.25	0.3619
	(75.05%)	Proposed w.o. $\mathfrak{s}$	18.75	11.35	0.3489	18.21	10.99	0.3475	18.24	11.11	0.2844
	(887.7ms)	Proposed w. $\mathfrak{s}$	18.46	10.55	**0.2231**	**17.94**	**10.28**	**0.2156**	**17.97**	**10.29**	**0.2274**
	SE(2)-MHA	OCP-STC	19.79	15.39	0.3330	20.36	22.64	0.3136	20.87	30.71	0.4271
	(98.50%)	Proposed w.o. $\mathfrak{s}$	18.26	11.41	0.5202	18.48	11.47	0.4662	18.92	11.78	0.4899
	(109.8ms)	Proposed w. $\mathfrak{s}$	**17.96**	**10.33**	0.2376	18.21	10.44	0.2317	18.57	10.61	0.2540

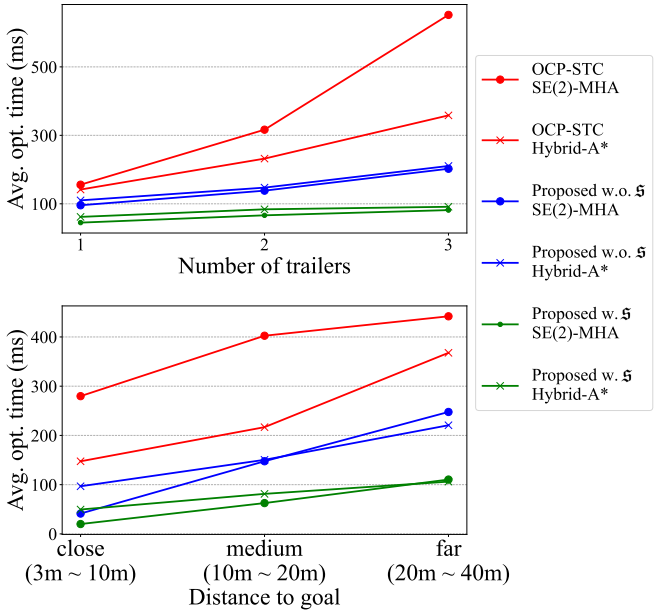


Fig. 11: Line graphs of average optimization time as a function of problem dimensions in the environment with a medium density of obstacles.

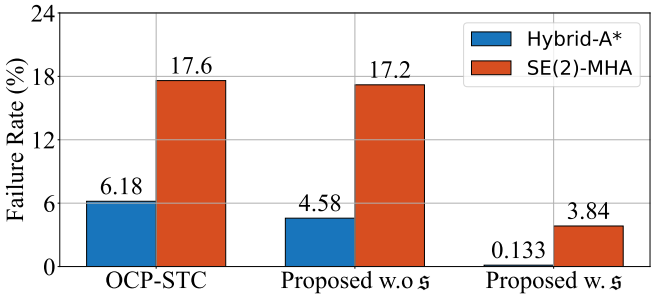


Fig. 12: Failure rate in the environment with a medium density of obstacles.

is all below 140 ms. This is attributed to the fact that it only searches in the $SE(2)$ space while utilizing the numerical integration approximation to obtain the trailers' trajectory. On the contrary, since Hybrid-A* has to do the search and collision detection in the complete state space, the success rate decreases with the increase of the number of trailers under the time limit of 5 seconds. It far exceeds SE(2)-MHA in terms of average time consumption, which is at least seven times more than the latter.

In Table V, Hybrid-A* + Proposed w. $\mathfrak{s}$ performs best in most cases. Hybrid-A* searches the space more thoroughly, the initial values obtained tend to be better than SE(2)-MHA. In most cases, Proposed w. $\mathfrak{s}$ initialized with SE(2)-MHA performs second best and has a smaller gap with Hybrid-A* + Proposed w. $\mathfrak{s}$. Although the trajectory length and trajectory duration of Proposed w.o. $\mathfrak{s}$ are slightly better than OCP-STC [7] in most cases, the average curvature of the tractor trajectories is the largest in many cases due to the inability to handle the singularity. Since the strategy of generating corridors for each vehicle may lead to a smaller feasible space containing less optimal solutions, OCP-STC [7] does not perform as well as the proposed methods in terms of trajectory length and trajectory duration.

To further evaluate the execution performance of trajectories, using Gazbeo [59], we model the robot used in the real-world experiments into the physical simulation shown in Fig. 10, recording more than one hundred trajectories

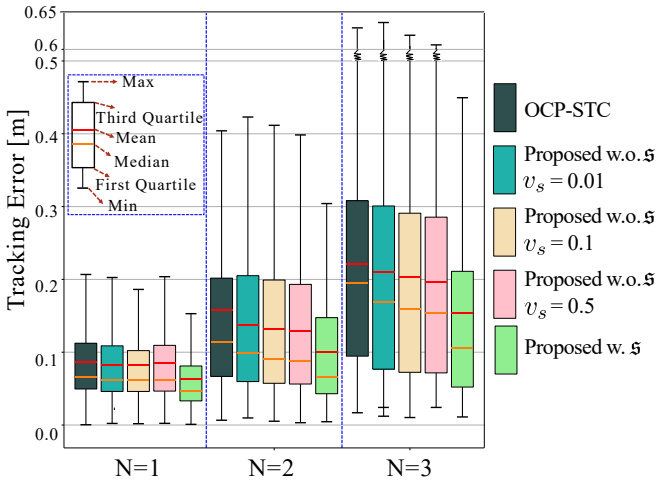


Fig. 13: The statistical analysis of tracking errors for each trajectory during their execution.

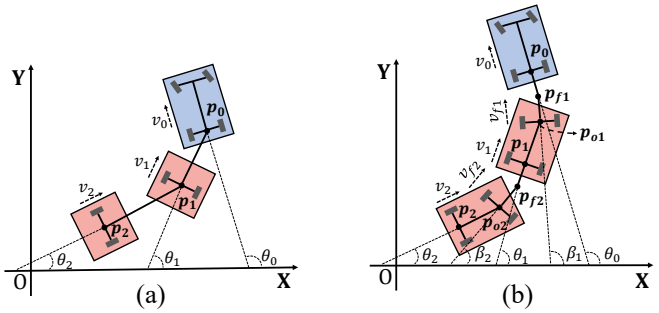


Fig. 14: Different tractor-trailer models studied in our work (a) and the work [30] (b).

for each of the robots with different numbers of trailers to perform the tracking error test as in the ablation experiment. The results are shown in Fig. 13. We can see that the trajectory tracking error decreases by almost 20% with the proposed method, comparing without using the slackened arc length $\mathfrak{s}$. In addition, compared to OCP-STC [7], our method also achieves lower tracking errors. We attribute this to the high-order continuous polynomial trajectory representation. Compared to methods employing discrete motion processes, our approach essentially guarantees the continuity of the state and finite-dimensional derivatives, making our trajectories closer to real physical motion processes, and thus easier to execute.

V. Discussion

A. Generalizability

In this work, we focus on tractor-trailer robots with trailers having only two passive wheels, as shown in Fig. 14(a). However, there are also tractor-trailer robots with trailers having four passive wheels in the real world, as shown in Fig. 14(b), which had been studied in the work [30]. In this setup, each trailer has four passive wheels, two axles, an on-axle hitch (points $\boldsymbol{p}_{o1}$ and $\boldsymbol{p}_{o2}$) and an off-axle hitch (points $\boldsymbol{p}_{f1}$ and $\boldsymbol{p}_{f2}$). This trailer model is more elaborate with

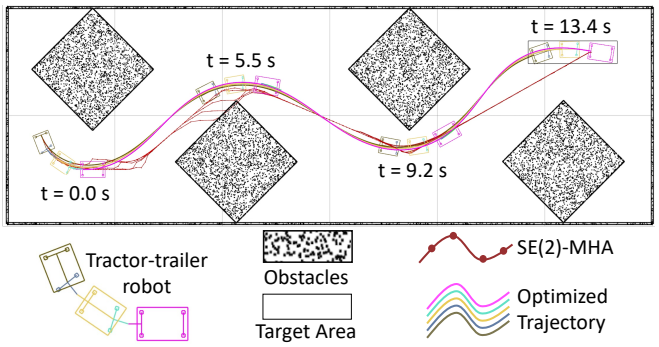


Fig. 15: Our pipeline can generalize to the more complex model studied in work [30].

the advantage of reducing the forces on the linkage, while making the motion of the trailer smoother and less likely to result in jackknife [4]. Despite such a more complex trailer model, with two more degrees of freedom compared to the model we studied, our method can still be quickly transferred on it, with only simple modifications to the optimization variables and constraints. Taking the example of a tractor with two identical trailers, the kinematic model for a robot similar to the one in the work [30] is:

$$\dot{x}_0(t) = v_0(t)\cos\theta_0(t), \tag{63}$$
$$\dot{y}_0(t) = v_0(t)\sin\theta_0(t), \tag{64}$$
$$\dot{v}_0(t) = a(t), \tag{65}$$
$$\dot{\theta}_0(t) = v_0(t)\frac{\tan\delta(t)}{L_0}, \tag{66}$$

$$\dot{\beta}_1(t) = \frac{v_0(t)\sin(\theta_0(t)-\beta_1(t))}{L_h} - \frac{L_t\dot{\theta}_0(t)\cos(\theta_0(t)-\beta_1(t))}{L_h}, \tag{67}$$
$$\dot{\theta}_1(t) = \frac{v_{f1}(t)\sin(\beta_1(t)-\theta_1(t))}{L_w}, \tag{68}$$
$$\dot{\beta}_2(t) = \frac{v_1(t)\sin(\theta_1(t)-\beta_2(t))}{L_h} - \frac{L_t\dot{\theta}_1(t)\cos(\theta_1(t)-\beta_2(t))}{L_h}, \tag{69}$$
$$\dot{\theta}_2(t) = \frac{v_{f2}(t)\sin(\beta_2(t)-\theta_2(t))}{L_w}, \tag{70}$$

where v_0, a, δ, L_0 are the velocity, longitudinal acceleration, steering angle, and wheelbase length of the tractor, respectively. L_t is the distance from the rear center of the previous car to the off-axle hitch. L_h is the distance from the off-axle hitch to the on-axle hitch in a trailer. L_w is the wheelbase length of a trailer. $v_{f1} = v_0\cos(\theta_0-\beta_1) + L_t\dot{\theta}_0\sin(\theta_0-\beta_1), v_1 = v_{f1}\cos(\beta_1-\theta_1), v_{f1} = v_1\cos(\theta_1-\beta_2) + L_t\dot{\theta}_1\sin(\theta_1-\beta_2)$.

Comparing the model we studied, by simply introducing just two new trajectories for β_1 and β_2, and modifying the equation constraints for kinematic constraints (34) to those

for kinematic constraints (67) $\sim$ (70) in the trajectory optimization problem, we are able to perform motion planning for this kinds of robot. As shown in Fig. 15, we performed a simulation experiment in the ROS environment. In this scenario, the robot is asked to traverse an area containing obstacles to reach the target area. The code targeting this complex trailer model is also open-sourced in a branch[2] in our Github repository as a reference for the community.

To further demonstrate the generalizability of our algorithm, we conduct real-world experiments on a larger-scale robot. we selected the *PIX KIT 2.0* chassis as the tractor, designing trailers similar to the one studied in work [30], as shown in Fig. 16 A. In this experiment, the kinematic constraints of the tractor are set to the maximum longitude velocity $v_{\text{mlon}} = 1.0m/s$, the maximum longitude acceleration $a_{\text{mlon}} = 0.8m/s^2$, the maximum latitude acceleration $a_{\text{mlat}} = 1.2m/s^2$, and maximum curvature $\kappa_{\text{max}} = 0.28m^{-1}$. To have smoother control commands, we also limit the maximum steering velocity $\dot{\delta}_{\text{max}} = 0.6rad/s$ of the tractor in trajectory optimization.

Since the control interfaces of the chassis are throttle, brake and steering angle, we modify the control output of the trajectory tracking controller to be acceleration and steering angle, interpolating the throttle-brake calibration table corresponding to speed and acceleration to obtain the final throttle or brake commands. As for the localization module, we place a LiDAR and run LIO on each trailer separately, aligning the world frame with the LIO running on the tractor. Knowing the absolute position of neighboring vehicles, we can use the cosine theorem to calculate the yaw angle β_1, β_2 of the wheel axle corresponding to the on-axle hitch.

We place e-bike, rockery, boxes as obstacles on a path of an industrial park and set up a cargo area, as shown in Fig. 16 B. The robot is requested to avoid obstacles to reach near the cargo area. After the staff load the cargo, it is required to continue traversing the obstacle area, as detailed in Fig. 16 C $\sim$ G. A more complete process can be found in the attached multimedia.

B. Tests under various uncertainty and noise

To further evaluate the impact of various uncertainties and noises on the proposed planner, we conduct simulation experiments in a $60m*10m$ unknown environment, as shown in Fig. 17. The tractor-trailer robot is configured with the same parameters as used in our real-world experiments and is equipped with an omnidirectional radar capable of sensing objects up to $10m$ long. After the planner outputs a trajectory, we use a MPC-based controller [58] to track it.

In the real world, the sources of uncertainty and noise are mainly caused by the perception and control modules. The uncertainty in the former mainly comes from the uncertainty in the perception of obstacles in the environment, and the uncertainty in the estimation of robot's state. While the uncertainty of the control module mainly originates from the uncertainty of the robot model. Thus, we add the noises from the following three aspects:

1) **Uncertainty in environment perception:** In our experiments, the robot's perception of the environment comes from the LiDAR. So we add noise to the distance scanned by the LiDAR: $\hat{d}_i = d_i + n_{di}, i = 1, 2, ..., N_l$, where d_i is the ground truth of the distance, N_l is the number of distances scanned by the LiDAR. $n_{di} \sim \mathcal{N}(0, \sigma_d^2)$, $\sigma_d > 0$ is the standard deviation of the normal distribution.
2) **Uncertainty in robot state perception:** Due to the deformability of the tractor-trailer robots, we impose noise on the yaw angle estimation of the trailers in the robot: $\hat{\theta}_i - \hat{\theta}_{i+1} = \theta_i - \theta_{i+1} + n_{\theta i}, i = 0, 1, ..., N_t$, where θ_i is the ground truth of the yaw angle, $\theta_0 = \hat{\theta}_0$ is the yaw angle of the tractor, $N_t + 1 = 3$ is the number of trailers, $n_{\theta i} \sim \mathcal{N}(0, \sigma_\theta^2)$.
3) **Uncertainty in robot model:** Referring to the experiments on model uncertainty in the work [12], the kinematic model with noise $\boldsymbol{n}_m$ used in this experiment is as follows:

$$\dot{\boldsymbol{s}} = f(\boldsymbol{s}, \boldsymbol{u}, \boldsymbol{n}_m), \tag{71}$$

$$\boldsymbol{s} = [x_0, y_0, v_0, \theta_0, \theta_1, \theta_2, \theta_3]^{\mathrm{T}}, \tag{72}$$

$$\boldsymbol{u} = [a, \delta]^{\mathrm{T}}, \tag{73}$$

where $[x_0, y_0]$ is the position of the rear center of the tractor, v_0 is the longitude velocity, a is the longitude acceleration, δ is the steer angle, θ_0 is yaw angle of the tractor, $\theta_i, i = \{1, 2, 3\}$ are yaw angles of the trailers, respectively. Here, the state transfer equation f with noise is defined as:

$$f(\boldsymbol{s}, \boldsymbol{u}, \boldsymbol{n}_m) = q(\boldsymbol{s}, \boldsymbol{u}) + N(\boldsymbol{s}, \boldsymbol{u}, \boldsymbol{n}_m), \tag{74}$$

$$\begin{aligned} q(\boldsymbol{s}, \boldsymbol{u}) = [&v_0 \cos\theta_0, v_0 \sin\theta_0, a, v_0 \tan\delta / L_w, \\ &v_0 \sin(\theta_0 - \theta_1)/L_t, \\ &v_1 \sin(\theta_1 - \theta_2)/L_t, \\ &v_2 \sin(\theta_2 - \theta_3)/L_t]^{\mathrm{T}}, \end{aligned} \tag{75}$$

where $v_1 = v_0 \cos(\theta_0 - \theta_1), v_2 = v_1 \cos(\theta_1 - \theta_2)$. L_w is the wheelbase length of the tractor, L_t is rigid link length of the trailer. q is the theoretical state transfer, and $N = \boldsymbol{n}_m q$ is the uncertainty term defined as proportional to the change in motion. $\boldsymbol{n}_m$ is a diagonal matrix representing the intensity of random noise, where each element satisfies a uniform distribution on $[h_1, h_2]$. Also, delays are introduced between the controller and actuator to better simulate real-world scenarios.

In this experiment, replanning is triggered if the original trajectory collides with a newly detected obstacle, or the tracking error exceeds $0.1m$. Robots hitting obstacles during movement are considered as failed attempts. The experiment cases are classified into four groups based on the levels of noise and delay: free uncertainty, low uncertainty, medium uncertainty, and high uncertainty. We refer to the datasheets

[2] https://github.com/Tracailer/Tracailer/tree/general_model

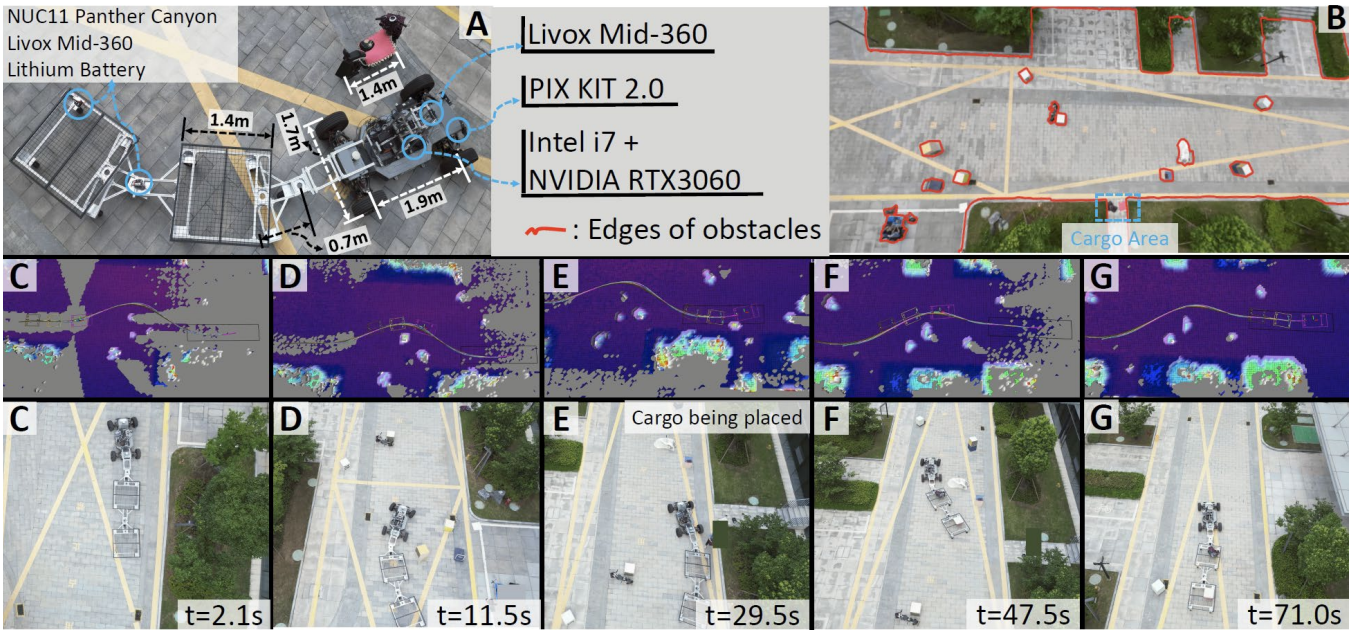


Fig. 16: Real-world experiments for the more complex model studied in work [30]. Figure A shows the configuration and size of the robot. Figure B shows the top view of the experimental scenario. Figure C $\sim$ G show the third-person viewpoints or corresponding views from Rviz tacken at different moments.

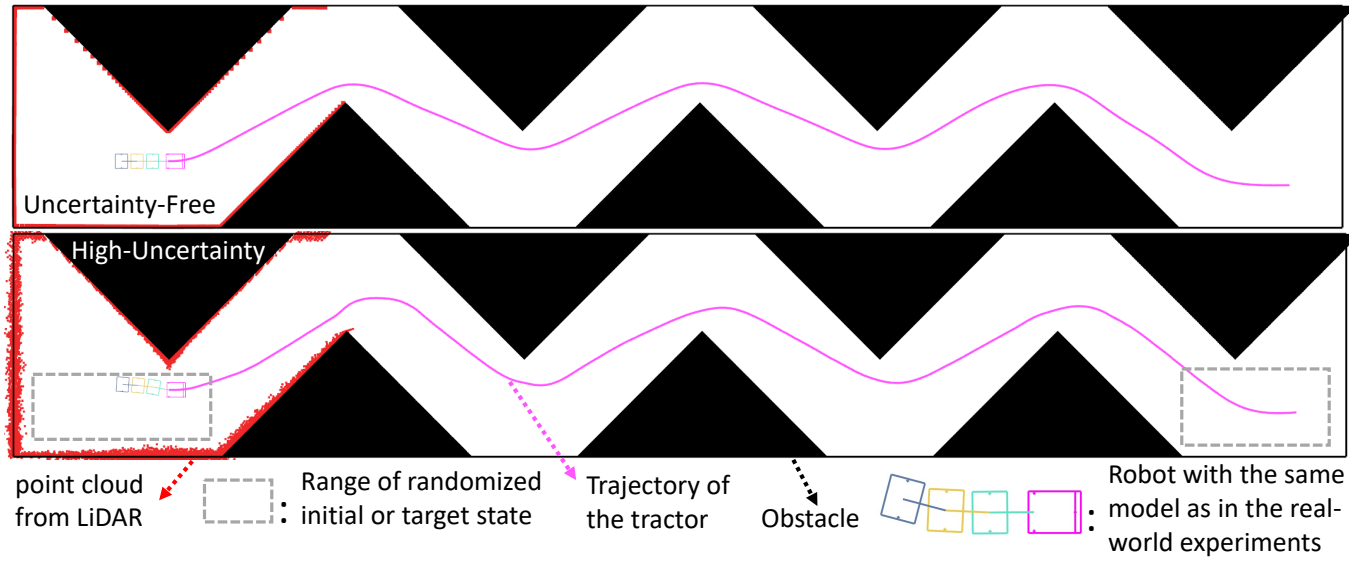


Fig. 17: Motion trajectory visualization under cases without and with high uncertainty.

TABLE VI: Statistics of Trajectory Dynamics Under Various Uncertainties.

Uncertainty	σ_d (m)	σ_θ (Deg)	Delay (ms)	Model	M.R	S.R (%)	M.TE (cm)	M.TL (m)	M.V (m/s)	M.SR (rad/s)
Free	0	0	0	[0, 0]	33	100	1.67	55.80	0.98	0.030
Low	0.05	1	0.02	[-0.05, 0.15]	34	100	2.73	55.94	0.93	0.033
Medium	0.1	2	0.05	[-0.075, 0.225]	43	100	5.07	56.18	0.87	0.042
High	0.15	4	0.07	[-0.1, 0.3]	248	84	9.59	58.45	0.45	0.060

and specifications of common LiDARs, such as *Livox MID-360*, Ouster [60] to select $\sigma_d = 5cm$ for the low uncertainty case. The specifications of the absolute encoders from *OMRON* and *HEIN LANZ* are referenced to set $\sigma_\theta = 1^\circ$ for the low uncertainty case. Finally, we set the delays and noise for the different cases by referring to the experiments on model

uncertainty in the work [12].

One hundred tests are conducted for each case. Moreover, success rate (S.R), mean number of replans (M.R), mean tracking error (M.TE), mean trajectory length (M.TL), mean velocity (M.V), and mean steering angle rate (M.SR) are calculated. The quantitative results are summarized in Table VI. Despite the presence of uncertainty, the robot still exhibits some resilience due to its high-frequency trajectory tracking controller and replanning capability, allowing it to navigate safely in unknown environments with 84% success rate even under high uncertainty. The noise intensity in this test environment actually exceeds that of the vast majority in the real world. Such rigorous simulations validate the robustness of our pipeline.

The results in the Table VI further indicate that high uncertainty leads to more frequent excessive tracking errors, which in turn results in a dramatic increase in the number of replans. On the other hand, replanning techniques allow the robot to eliminate accumulated tracking errors, thus enhancing its robustness against uncertainty. However, the increase in uncertainty still has some other impact. As shown by the increase in mean tracking error and steering rate, it becomes more difficult to track the trajectory. Second, smaller mean velocities and longer trajectories imply reduced optimality. In the future, we will incorporate the potential uncertainties in the controllers and the proposed planner, with the expectation of further increasing the practical applicability and the resilience of the system to uncertainty.

VI. Conclusion

In this paper, we propose a new trajectory representation for tractor-trailer robot. Based on it, we formulate the motion planning problem as an optimization problem which can be simplified and efficiently solved by numerical algorithm. Using SDF as environment representation and proposing a multi-terminal path searching method, we achieved more efficient trajectory generation for tractor-trailer robots in unstructured environments.

In the future, we will further improve the adaptability of our algorithms to cope with more dynamic environments. In addition, integrating technologies from machine learning or graphics to achieve smarter autonomous navigation, such as allowing tractor-trailer robots to acquire semantic information using GRB cameras to have more interaction with real-world environments, or minimizing sweep volume of the trajectory for reducing collision risk [61] are also promising directions to explore. Through these efforts, we believe that the application prospects for tractor-trailer robots in various transportation and logistics tasks will become even more expansive.

References

[1] H. R. Kam, S.-H. Lee, T. Park, and C.-H. Kim, "Rviz: a toolkit for real domain data visualization," *Telecommunication Systems*, vol. 60, pp. 337–345, 2015.

[2] P. Rouchon, M. Fliess, J. Lévine, and P. Martin, "Flatness, motion planning and trailer systems," in *Proceedings of 32nd IEEE Conference on Decision and Control*. IEEE, 1993, pp. 2700–2705.

[3] B. Li, T. Acarman, Y. Zhang, L. Zhang, C. Yaman, and Q. Kong, "Tractor-trailer vehicle trajectory planning in narrow environments with a progressively constrained optimal control approach," *IEEE Transactions on Intelligent Vehicles*, vol. 5, no. 3, pp. 414–425, 2019.

[4] M. Beglini, L. Lanari, and G. Oriolo, "Anti-jackknifing control of tractor-trailer vehicles via intrinsically stable mpc," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 8806–8812.

[5] O. Ljungqvist, N. Evestedt, M. Cirillo, D. Axehill, and O. Holmer, "Lattice-based motion planning for a general 2-trailer system," in *2017 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2017, pp. 819–824.

[6] R. Lattarulo, J. Pérez, and J. Murgoitio, "Rrt trajectory planning approach for automated semi-trailer truck parking," in *2022 IEEE International Conference on Vehicular Electronics and Safety (ICVES)*. IEEE, 2022, pp. 1–7.

[7] H. Cen, B. Li, T. Acarman, Y. Zhang, Y. Ouyang, and Y. Dong, "Optimization-based maneuver planning for a tractor-trailer vehicle in complex environments using safe travel corridors," in *2021 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2021, pp. 974–979.

[8] D. B. Muralidhara, O. Gehring, H. G. Bock, and L. Wirsching, "Scenario analysis for optimized trajectories of a truck-trailer model utilizing coupled dynamics," in *2022 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2022, pp. 708–717.

[9] K. Bergman, O. Ljungqvist, and D. Axehill, "Improved path planning by tightly combining lattice-based path planning and optimal control," *IEEE Transactions on Intelligent Vehicles*, vol. 6, no. 1, pp. 57–66, 2020.

[10] N. Ito, H. Okuda, S. Inagaki, and T. Suzuki, "Configuration-aware model predictive motion planning in narrow environment for autonomous tractor-trailer mobile robot," in *IECON 2021–47th Annual Conference of the IEEE Industrial Electronics Society*. IEEE, 2021, pp. 1–7.

[11] D. Mellinger and V. Kumar, "Minimum snap trajectory generation and control for quadrotors," in *2011 IEEE international conference on robotics and automation*. IEEE, 2011, pp. 2520–2525.

[12] Z. Han, Y. Wu, T. Li, L. Zhang, L. Pei, L. Xu, C. Li, C. Ma, C. Xu, S. Shen, *et al.*, "An efficient spatial-temporal trajectory planner for autonomous vehicles in unstructured environments," *IEEE Transactions on Intelligent Transportation Systems*, 2023.

[13] Z. Han, M. Tian, and F. Gao, "Trajectory generation for vehicle with stable time." [Online]. Available: https://mengze3.github.io/files/technical_report_.pdf

[14] D. Dolgov, S. Thrun, M. Montemerlo, and J. Diebel, "Practical search techniques in path planning for autonomous driving," *Ann Arbor*, vol. 1001, no. 48105, pp. 18–80, 2008.

[15] Z.-J. Liu, P. Huang, J.-L. Huang, and J.-L. Que, "Path planning for tractor-trailer mobile robot based on heuristic genetic algorithm," in *Proceedings of 2004 International Conference on Machine Learning and Cybernetics (IEEE Cat. No. 04EX826)*, vol. 2. IEEE, 2004, pp. 1119–1124.

[16] F. Sun, Y. Huang, J. Yuan, Y. Kang, and F. Ma, "A compound prm method for path planning of the tractor-trailer mobile robot," in *2007 IEEE International Conference on Automation and Logistics*. IEEE, 2007, pp. 1880–1885.

[17] J. Leu, Y. Wang, M. Tomizuka, and S. Di Cairano, "Improved a-search guided tree for autonomous trailer planning," in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2022, pp. 7190–7196.

[18] J. Cheng, Y. Zhang, and Z. Wang, "Motion planning algorithm for tractor-trailer mobile robot in unknown environment," in *2012 8th International Conference on Natural Computation*. IEEE, 2012, pp. 1050–1055.

[19] S. Karaman and E. Frazzoli, "Sampling-based algorithms for optimal motion planning," *The international journal of robotics research*, vol. 30, no. 7, pp. 846–894, 2011.

[20] D. J. Webb and J. Van Den Berg, "Kinodynamic rrt*: Asymptotically optimal motion planning for robots with linear dynamics," in *2013 IEEE international conference on robotics and automation*. IEEE, 2013, pp. 5054–5061.

[21] A. C. Manav and I. Lazoglu, "A novel cascade path planning algorithm for autonomous truck-trailer parking," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 7, pp. 6821–6835, 2021.

[22] X. Zhang, J. Eck, and F. Lotz, "A path planning approach for tractor-trailer system based on semi-supervised learning," in *2022 IEEE 25th International Conference on Intelligent Transportation Systems*

(ITSC). IEEE, 2022, pp. 3549–3555.

[23] J. Cheng, Y. Zhang, and Z. Wang, "Backward tracking control of mobile robot with one trailer via fuzzy line-of-sight method," in *2009 Sixth International Conference on Fuzzy Systems and Knowledge Discovery*, vol. 4. IEEE, 2009, pp. 66–70.

[24] O. Ljungqvist, N. Evestedt, D. Axehill, M. Cirillo, and H. Pettersson, "A path planning and path-following control framework for a general 2-trailer with a car-like tractor," *Journal of field robotics*, vol. 36, no. 8, pp. 1345–1377, 2019.

[25] J. Dahlmann, A. Völz, M. Lukassek, and K. Graichen, "Local predictive optimization of globally planned motions for truck-trailer systems," *IEEE Transactions on Control Systems Technology*, 2023.

[26] J. Dahlmann, A. Völz, T. Szabo, and K. Graichen, "Trajectory optimization for truck-trailer systems based on predictive path-following control," in *2022 IEEE Conference on Control Technology and Applications (CCTA)*. IEEE, 2022, pp. 37–42.

[27] D. Dolgov, S. Thrun, M. Montemerlo, and J. Diebel, "Path planning for autonomous vehicles in unknown semi-structured environments," *The international journal of robotics research*, vol. 29, no. 5, pp. 485–501, 2010.

[28] A. Mohamed, J. Ren, H. Lang, and M. El-Gindy, "Optimal collision free path planning for an autonomous articulated vehicle with two trailers," in *2017 IEEE International Conference on Industrial Technology (ICIT)*. IEEE, 2017, pp. 860–865.

[29] Y. Wang, Z. Wang, J. Wang, R. Guo, and P. Xiao, "Pi2-cma based trajectory planning method for the tractor-trailer vehicle," in *2023 IEEE 6th International Conference on Pattern Recognition and Artificial Intelligence (PRAI)*. IEEE, 2023, pp. 1230–1235.

[30] H. Zhao, W. Chen, S. Zhou, F. Zheng, and Y.-H. Liu, "Localization and motion planning of industrial tractor–trailers vehicles," *IEEE Transactions on Control Systems Technology*, vol. 31, no. 6, pp. 2928–2936, 2023.

[31] H. Zhao, W. Chen, S. Zhou, Z. Liu, F. Zheng, and Y.-H. Liu, "Online trajectory planning for an industrial tractor towing multiple full trailers," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 6089–6095.

[32] B. Li, Y. Zhang, T. Acarma, Q. Kong, and Y. Zhang, "Trajectory planning for a tractor with multiple trailers in extremely narrow environments: A unified approach," in *2019 international conference on robotics and automation (ICRA)*. IEEE, 2019, pp. 8557–8562.

[33] M. Zhao, T. Shen, F. Wang, G. Yin, Z. Li, and Y. Zhang, "Aptenplanner: Autonomous parking of semi-trailer train in extremely narrow environments," *IEEE Transactions on Intelligent Transportation Systems*, 2023.

[34] X. Ruan, Y. Huang, Y. Wang, Z. Fu, L. Xiong, and Z. Yu, "An efficient trajectory-planning method with a reconfigurable model for any tractor-trailer vehicle," *IEEE Transactions on Transportation Electrification*, vol. 9, no. 2, pp. 3360–3374, 2022.

[35] H. Zhuang, Q. Shen, Y. Qian, W. Yuan, C. Wang, and M. Yang, "Fast bidirectional motion planning for self-driving general n-trailers vehicle maneuvering in narrow space," *IEEE Open Journal of Intelligent Transportation Systems*, 2023.

[36] B. Li, L. Li, T. Acarman, Z. Shao, and M. Yue, "Optimization-based maneuver planning for a tractor-trailer vehicle in a curvy tunnel: A weak reliance on sampling and search," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 706–713, 2021.

[37] R. M. Murray and S. S. Sastry, "Nonholonomic motion planning: Steering using sinusoids," *IEEE Transactions on Automatic Control*, vol. 38, no. 5, pp. 700–716, 1993.

[38] V. Deligiannis, G. Davrazos, S. Manesis, and T. Arampatzis, "Flatness conservation in the n-trailer system equipped with a sliding kingpin mechanism," *Journal of Intelligent and Robotic Systems*, vol. 46, pp. 151–162, 2006.

[39] M. Zhang, C. Xu, F. Gao, and Y. Cao, "Trajectory optimization for 3d shape-changing robots with differential mobile base," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2023, pp. 10 104–10 110.

[40] A. Kelly and B. Nagy, "Reactive nonholonomic trajectory generation via parametric optimal control," *The International Journal of Robotics Research*, vol. 22, no. 7-8, pp. 583–601, 2003.

[41] L. Xu, K. Chai, Z. Han, H. Liu, C. Xu, Y. Cao, and F. Gao, "An efficient trajectory planner for car-like robots on uneven terrain," in *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2023, pp. 2853–2860.

[42] I. Bae, J. Moon, J. Jhung, H. Suk, T. Kim, H. Park, J. Cha, J. Kim, D. Kim, and S. Kim, "Self-driving like a human driver instead of a robocar: Personalized comfortable driving experience for autonomous vehicles," *arXiv preprint arXiv:2001.03908*, 2020.

[43] Z. Wang, X. Zhou, C. Xu, and F. Gao, "Geometrically constrained trajectory optimization for multicopters," *IEEE Transactions on Robotics*, vol. 38, no. 5, pp. 3259–3278, 2022.

[44] J. Leslie, "On a differential structure for the group of diffeomorphisms," *Topology*, vol. 6, no. 2, pp. 263–271, 1967.

[45] R. T. Rockafellar, "Augmented lagrange multiplier functions and duality in nonconvex programming," *SIAM Journal on Control*, vol. 12, no. 2, pp. 268–285, 1974.

[46] H. Wang, H. Li, B. Zhou, F. Gao, and S. Shen, "Impact-aware planning and control for aerial robots with suspended payloads," *IEEE Transactions on Robotics*, 2024.

[47] L. E. Dubins, "On curves of minimal length with a constraint on average curvature, and with prescribed initial and terminal positions and tangents," *American Journal of mathematics*, vol. 79, no. 3, pp. 497–516, 1957.

[48] F. Zhu, Y. Ren, F. Kong, H. Wu, S. Liang, N. Chen, W. Xu, and F. Zhang, "Swarm-lio: Decentralized swarm lidar-inertial odometry," in *2023 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2023, pp. 3254–3260.

[49] L. Yin, F. Zhu, Y. Ren, F. Kong, and F. Zhang, "Decentralized swarm trajectory generation for lidar-based aerial tracking in cluttered environments," in *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2023, pp. 9285–9292.

[50] W. Xu, Y. Cai, D. He, J. Lin, and F. Zhang, "Fast-lio2: Fast direct lidar-inertial odometry," *IEEE Transactions on Robotics*, vol. 38, no. 4, pp. 2053–2073, 2022.

[51] P. Fankhauser, M. Bloesch, and M. Hutter, "Probabilistic terrain mapping for mobile robots with uncertain localization," *IEEE Robotics and Automation Letters*, vol. 3, no. 4, pp. 3019–3026, 2018.

[52] T. Miki, L. Wellhausen, R. Grandia, F. Jenelten, T. Homberger, and M. Hutter, "Elevation mapping for locomotion and navigation using gpu," in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2022, pp. 2273–2280.

[53] M. Pauly, M. Gross, and L. P. Kobbelt, "Efficient simplification of point-sampled surfaces," in *IEEE Visualization, 2002. VIS 2002.* IEEE, 2002, pp. 163–170.

[54] P. F. Felzenszwalb and D. P. Huttenlocher, "Distance transforms of sampled functions," *Theory of computing*, vol. 8, no. 1, pp. 415–428, 2012.

[55] J. A. E. Andersson, J. Gillis, G. Horn, J. B. Rawlings, and M. Diehl, "CasADi – A software framework for nonlinear optimization and optimal control," *Mathematical Programming Computation*, vol. 11, no. 1, pp. 1–36, 2019.

[56] A. Wächter and L. T. Biegler, "On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming," *Mathematical programming*, vol. 106, pp. 25–57, 2006.

[57] R. Fourer, D. M. Gay, and B. W. Kernighan, "Ampl: A mathematical programming language," *Management Science*, vol. 36, no. 5, pp. 519–554, 1990.

[58] K. R. Muske and J. B. Rawlings, "Model predictive control with linear models," *AIChE Journal*, vol. 39, no. 2, pp. 262–287, 1993.

[59] N. Koenig and A. Howard, "Design and use paradigms for gazebo, an open-source multi-robot simulator," in *2004 IEEE/RSJ international conference on intelligent robots and systems (IROS)(IEEE Cat. No. 04CH37566)*, vol. 3. Ieee, 2004, pp. 2149–2154.

[60] O. M.-R. H.-R. Imaging, "Os1 mid-range high-resolution imaging lidar," *San Francisco: Ouster*, 2021.

[61] T. Hu, S. Yuan, R. Bai, X. Xu, Y. Liao, F. Liu, and L. Xie, "Swept volume-aware trajectory planning and mpc tracking for multi-axle swerve-drive amrs," *arXiv preprint arXiv:2412.16875*, 2024.