Optimization-Based Maneuver Planning for a Tractor-Trailer Vehicle in a Curvy Tunnel: A Weak Reliance on Sampling and Search

Bai Li, *Member, IEEE*, Li Li, *Fellow, IEEE*, Tankut Acarman, *Member, IEEE*, Zhijiang Shao, and Ming Yue, *Member, IEEE*

***Abstract*—This study is focused on the maneuver planning problem for a tractor-trailer vehicle in a curvy and tiny tunnel. Due to the curse of dimensionality, the prevalent sampling-and- search-based planners used to handle a rigid-body vehicle well become less efficient when the trailer number grows or when the tunnel narrows. This fact also has impacts on an optimization-based planner if it counts on a sampling-and-search-based initial guess to warm-start. We propose an optimization-based maneuver planner that weakly relies on the sampling and search, hoping to get rid of the curse of dimensionality and thus find optima rapidly. The proposed planner comprises three stages: stage 1 identifies the homotopy class via A$*$ search in a 2D grid map; stage 2 recovers the kinematic feasibility with softened intermediate problems iteratively solved; stage 3 finds an optimum that strictly satisfies the nominal collision-avoidance constraints. Optimization-based planners are commonly known to run slowly, but this work shows that they have obvious advantages over the prevalent sampling-and-search-based planners when the solution space dimension is high and/or the constraints are harsh.**

***Index Terms*—Optimization and optimal control, nonholonomic motion planning, tractor-trailer vehicle.**

I. Introduction

A TRACTOR-TRAILER vehicle system is constituted by a towing tractor equipped with underactuated trailers [1]. A tractor-trailer vehicle sweeps a smaller area than the same-length rigid-body vehicle when turning around [2], [3]. This property enables its agile movement in tiny environments, thus yielding wide applications in agriculture, logistics, and military [4]–[6].

Manuscript received September 1, 2021; accepted November 18, 2021. Date of publication December 1, 2021; date of current version December 14, 2021. This letter was recommended for publication by Associate Editor T. Bandyopadhyay and Editor H. Kurniawati upon evaluation of the reviewers' comments. This work was supported in part by the National Key Research and Development Program of China under Grant 2020AAA0108104, in part by the National Natural Science Foundation of China under Grant 62103139, in part by the Fundamental Research Funds for the Central Universities under Grant 531118010509, and in part by the Natural Science Foundation of Hunan Province, China under Grant 2021JJ40114. *(Corresponding author: Li Li.)*

Bai Li is with the College of Mechanical and Vehicle Engineering, Hunan University, Changsha 410082, China (e-mail: libaioutstanding@163.com).

Li Li is with the Department of Automation BNRist, Tsinghua University, Beijing 100084, China (e-mail: li-li@tsinghua.edu.cn).

Tankut Acarman is with the Department of Computer Engineering, Galatasaray University, Istanbul 34349, Turkey (e-mail: acarmant@gmail.com).

Zhijiang Shao is with the College of Control Science and Engineering, Zhejiang University, Hangzhou 310027, China (e-mail: szj@zju.edu.cn).

Ming Yue is with the School of Automotive Engineering, Dalian University of Technology, Dalian 116622, China (e-mail: yueming@dlut.edu.cn).

Digital Object Identifier 10.1109/LRA.2021.3131693

Driving a tractor-trailer vehicle manually is challenging even for an experienced driver because the vehicle kinematics is against human intuitions [7]. The situation is more difficult when the trailer number increases. Maneuver planning is a critical aspect of an autonomous tractor-trailer vehicle system. High-quality maneuver planning results can largely facilitate the closed-loop tracking process, especially for backward maneuvers [8]–[10]. This study is focused on the maneuver planning task in a tiny tunnel-like scenario.

A. Related Works

Predominant maneuver planners suitable for a tractor-trailer vehicle are the sampling-and-search-based methods, which abstract the configuration space as a graph with nodes and edges, where the edges denote the sampled state/control primitives [11]–[13]. Search operations are typically conducted by Dijkstra [14], A$*$ [15], [16], and dynamic programming [17]. Since a tractor-trailer vehicle is typically an underactuated nonholonomic system, sampling in the state space easily involves kinematic infeasibility. That is why the existing methods mostly sample in the control space. However, any feasible solution may not exist among the finite number of sampled control primitives when the environment is highly constrained, which makes a sampling-and-search-based planner incomplete. The incompleteness becomes more serious with the growth of trailer number. This is because the drivable area narrows as per the complicated kinematics.

Optimization-based planners are also deployed to handle the concerned task. An optimization-based method describes a planning task as an optimal control problem (OCP) and then solves it analytically or numerically. Analytical OCP methods [18], [19] only deal with easy cases in sparse scenarios with few obstacles, while numerical OCP methods are promising to handle complicated cases [1]. Solving an OCP numerically refers to discretizing the OCP into a nonlinear program (NLP) before solving it via a gradient-based optimizer. Numerical OCP methods heavily rely on the initial guess to identify a homotopy class because the embedded gradient-based NLP solver cannot achieve global convergence. It is a natural idea to find a coarse path/trajectory via sampling and search to identify the homotopy class, and then use it to warm-start the numerical OCP solution process. Liu *et al.* [4] searched for collision-free waypoints and then connected Reeds-Shepp primitives between adjacent waypoints to form an initial guess for numerical OCP. Bergman *et al.* [20] divided a searched path into segments and solved several subtle OCPs with the two-point boundary

This work is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 License. For more information, see https://creativecommons.org/licenses/by-nc-nd/4.0/

conditions of each segment. Li *et al.* [10] proposed an extended hybrid A$*$ to find a coarse path and then solved a series of OCPs with increasing sizes of obstacles toward their nominal ones. None of these studies attempts to simplify the intractably scaled collision-avoidance constraints, thus resulting in a long runtime. Since the homotopy class is identified via the coarse path/trajectory, a spatio-temporal corridor can be paved to separate the ego vehicle from the surrounding obstacles. Within-corridor constraints are simple and tractably scaled, thus can be used to replace the nominal collision-avoidance constraints [21]. This idea has been realized in on-road cruising [22] and off-road parking [23] for a rigid-body vehicle. However, for a multi-body vehicle, different vehicle bodies may lie in different parts of the spatio-temporal corridor. Although one may allow each vehicle body to have its own spatio-temporal corridor [24], the correctness of corridor assignment of corridor relies on a kinematically feasible reference trajectory. An OCP with within-corridor constraints easily becomes infeasible if the reference trajectory is of low quality. Similar to the corridor-based planners, a trust-region-based planner [25] is also subjected to dependence on the sampling-and-search result. Overall, most of the existing numerical-OCP-based planners excessively rely on high-accuracy sampling and search, which may be inapplicable when the planning task is really difficult. Instead of involving sampling and search, one may build an iterative optimization framework wherein the difficulties are gradually resolved. Li and Shao [26] proposed an optimization-based method that finds feasible, suboptimal, and optimal solutions in a sequence, but the sequential planning process suffers from intermediate failure risks. Li *et al.* [10] first discarded the challenging constraints in the nominal OCP, then adaptively added them back. This method runs slowly because the division of the entire difficulties is not concise. Therefore, the existing optimization-based planners that try to divide and conquer the entire constraints are imperfect.

B. Contributions

This study aims to plan fast, optimal, and stable maneuvers for a generic tractor-trailer vehicle in a curvy and tiny tunnel. The proposed planner relies weakly on the sampling-and-search process; instead, it builds a three-stage architecture to divide and conquer the difficulties based on a numerical-OCP-based method. Concretely, Stage 1 finds a collision-free but kinematically infeasible coarse path; Stage 2 attempts to recover kinematic feasibility while Stage 3 finds an optimum. The main contributions lie in Stages 2 and 3. Stage 2 is featured by being fast and always feasible; Stage 3 is featured by being lightweight, optimal, and satisfying the nominal collision-avoidance constraints strictly. The entire three-stage planner runs faster than the existing methods in this research area.

II. Problem Statement

In this section, a standard maneuver planning scheme for a tractor-trailer vehicle is formulated as an OCP, which consists of a cost function and three types of constraints. Detailed formulations are presented as follows.

A. Vehicle Kinematics

A tractor-trailer vehicle system is formed by a tractor towing $\mathrm{N_V}$ trailers. For convenience, a unique ID is assigned to each unit of the system. The tractor is indexed as 0, and the *i*th trailer

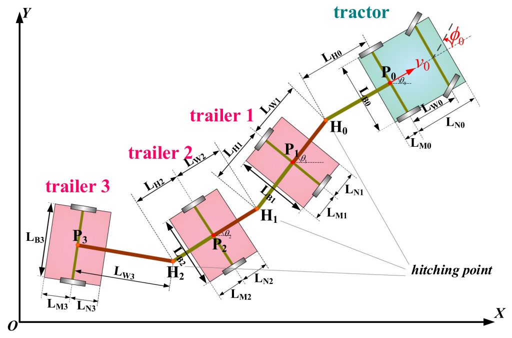

Fig. 1. A tractor towing 3 trailers, i.e., $\mathrm{N_V} = 3$. Visibility of this figure can be improved if one zooms it in.

is indexed as i $(i = 1, \ldots, \mathrm{N_V})$. The kinematic principle of the tractor is described by the well-known bicycle model:

$$\frac{\mathrm{d}}{\mathrm{d}t}\begin{bmatrix} x_0(t) \\ y_0(t) \\ v_0(t) \\ \phi_0(t) \\ \theta_0(t) \end{bmatrix} = \begin{bmatrix} v_0(t)\cdot\cos\theta_0(t) \\ v_0(t)\cdot\sin\theta_0(t) \\ a_0(t) \\ \omega_0(t) \\ v_0(t)\cdot\tan\phi_0(t)/\mathrm{L_{W0}} \end{bmatrix}, t \in [0, T]. \quad (1)$$

T denotes the completion time of the moving process, which is not known *a priori*. Let us denote the mid-point of the rear wheel axle as $P_0 = (x_0, y_0)$, θ_0 the orientation angle, v_0 the velocity, a_0 the acceleration, ϕ_0 the front-wheel steering angle, ω_0 the steering angle rate, and $\mathrm{L_{W0}}$ the wheelbase. As depicted in Fig. 1, $(\mathrm{L_{N0}} - \mathrm{L_{W0}})$ is the front overhang length, $\mathrm{L_{M0}}$ is the rear overhang length, and $\mathrm{L_{B0}}$ is the tractor width.

Bounds are imposed on some state/control profiles:

$$-\mathrm{a_{max}} \le a_0(t) \le \mathrm{a_{max}}, \quad (2a)$$

$$-\mathrm{v_{max}} \le v_0(t) \le \mathrm{v_{max}}, \quad (2b)$$

$$-\Phi_{\max} \le \phi_0(t) \le \Phi_{\max}, \quad (2c)$$

$$-\Omega_{\max} \le \omega_0(t) \le \Omega_{\max}, t \in [0, T] \quad (2d)$$

where $\mathrm{a_{max}}$, $\mathrm{v_{max}}$, $\Phi_{\max}$, and $\Omega_{\max}$ are amplitude parameters.

The tractor is connected with trailer 1 at the hitching point $H_0 = (x_0^{\mathrm{H}}, y_0^{\mathrm{H}})$ situated along the longitudinal axle of the tractor (Fig. 1):

$$x_0^{\mathrm{H}}(t) = x_0(t) - \mathrm{L_{H0}}\cdot\cos\theta_0(t),$$
$$y_0^{\mathrm{H}}(t) = y_0(t) - \mathrm{L_{H0}}\cdot sin\theta_0(t), \quad (3a)$$

where $\mathrm{L_{H0}}$ denotes the hitching offset. $P_1 = (x_1, y_1)$ is the mid-point along the wheel axle of trailer 1, and the orientation angle is denoted by θ_1. The coordinates of P_1 are identified via

$$x_1(t) = x_0^{\mathrm{H}}(t) - \mathrm{L_{W1}}\cdot\cos\theta_1(t),$$
$$y_1(t) = y_0^{\mathrm{H}}(t) - \mathrm{L_{W1}}\cdot sin\theta_1(t). \quad (3b)$$

For the generalization purpose, the mid-point coordinates along the wheel axle of trailer i are defined by

$$\begin{cases} x_i(t) = x_{i-1}(t) - \mathrm{L}_{\mathrm{W}i}\cdot\cos\theta_i(t) - \mathrm{L}_{\mathrm{H}(i-1)}\cdot\cos\theta_{i-1}(t) \\ y_i(t) = y_{i-1}(t) - \mathrm{L}_{\mathrm{W}i}\cdot\sin\theta_i(t) - \mathrm{L}_{\mathrm{H}(i-1)}\cdot\sin\theta_{i-1}(t) \end{cases},$$
$$i = 1, \ldots, \mathrm{N_V}. \quad (4)$$

According to [10], $\theta_i(t)$ is given as follows:

$$\frac{\mathrm{d}\theta_i(t)}{\mathrm{d}t} = \frac{v_{i-1} \cdot \sin\left(\theta_{i-1}(t) - \theta_i(t)\right)}{\mathrm{L}_{\mathrm{W}i}} - \frac{\mathrm{L}_{\mathrm{H}(i-1)} \cdot \cos\left(\theta_{i-1}(t) - \theta_i(t)\right) \cdot \mathrm{d}\theta_{i-1}(t)}{\mathrm{L}_{\mathrm{W}i} \cdot \mathrm{d}t}, i = 1, \ldots, \mathrm{N_V}. \tag{5}$$

Also, we have

$$v_i(t) = v_{i-1}(t) \cdot \cos\left(\theta_{i-1}(t) - \theta_i(t)\right) + \mathrm{L}_{\mathrm{H}(i-1)} \cdot \sin\left(\theta_{i-1}(t) - \theta_i(t)\right) \cdot \frac{\mathrm{d}\theta_{i-1}(t)}{\mathrm{d}t}, i = 1, \ldots, \mathrm{N_V}. \tag{6}$$

Jackknife refers to the situation that the hitch angle between adjacent vehicle bodies grows such that the vehicle folds on itself, thus making the entire vehicle uncontrollable [27]. The orientation angle difference between adjacent bodies i and $(i-1)$ should be bounded to prevent jackknife:

$$|\theta_i(t) - \theta_{i-1}(t)| \leq \pi/2 - buffer, i = 1, \ldots, \mathrm{N_V}. \tag{7}$$

Herein, $buffer \geq 0$ is a user-specified parameter to prevent the occurrence of a jackknife.

B. Two-Point Boundary Constraints

Boundary constraints specify the vehicle's configurations at $t = 0$ and T. The following constraints are used to specify the tractor status at $t = 0$:

$$[x_0(0), y_0(0), \theta_0(0), v_0(0), \phi_0(0), a_0(0), \omega_0(0)] = [\mathrm{x}_0^{\mathrm{init}}, \mathrm{y}_0^{\mathrm{init}}, \theta_0^{\mathrm{init}}, \mathrm{v}_0^{\mathrm{init}}, \phi_0^{\mathrm{init}}, \mathrm{a}_0^{\mathrm{init}}, \mathrm{w}_0^{\mathrm{init}}]. \tag{8a}$$

The initial status of each trailer is identified via

$$[\theta_i(0), v_i(0)] = [\theta_i^{\mathrm{init}}, \mathrm{v}_i^{\mathrm{init}}], i = 1, \ldots, \mathrm{N_V}. \tag{8b}$$

Parameters $\mathrm{x}_0^{\mathrm{init}}$, $\mathrm{y}_0^{\mathrm{init}}$, θ_0^{init}, etc. reflect the vehicle's driving status at $t = 0$. Notably, the initial location of trailer i, namely $(x_i(0), y_i(0))$,is not mentioned because it can be uniquely identified according to (8a), (8b), and (4).

The end-point constraints are defined similarly:

$$[x_0(T), y_0(T), \theta_0(T), v_0(T), \phi_0(T), a_0(T), \omega_0(T)] = [\mathrm{x}_0^{\mathrm{end}}, \mathrm{y}_0^{\mathrm{end}}, \theta_0^{\mathrm{end}}, \mathrm{v}_0^{\mathrm{end}}, \phi_0^{\mathrm{end}}, \mathrm{a}_0^{\mathrm{end}}, \mathrm{w}_0^{\mathrm{end}}]. \tag{9a}$$

$$[\theta_i(T), v_i(T)] = [\theta_i^{\mathrm{end}}, \mathrm{v}_i^{\mathrm{end}}], i = 1, \ldots, \mathrm{N_V}. \tag{9b}$$

C. Collision-Avoidance Constraints

This study particularly focuses on tunnel-like environments. $\mathrm{N_{OBS}}$ convex polygonal obstacles are used to decorate a tunnel. The tractor-trailer vehicle should not collide with each of the obstacles at $\forall t \in [0, T]$.

Without loss of generality, let us consider the collision-avoidance constraint between the jth obstacle and the *i*th vehicle body $(j = 1, \ldots, \mathrm{N_{OBS}}, i = 0, \ldots, \mathrm{N_V})$. Suppose that the four vertexes of the *i*th vehicle body are denoted as A_i, B_i, C_i, and D_i, and that the *j*th obstacle has $\mathrm{N}_j^{\mathrm{ver}}$ vertexes, which are denoted as $\mathrm{V}_{j1}, \ldots, \mathrm{V}_{j\mathrm{N}_j^{\mathrm{ver}}}$. Any collision always begins with the moment when an obstacle vertex hits the vehicle body or a vehicle body vertex hits the obstacle. If the following two conditions are required, then collisions do not occur: 1) A_i, B_i, C_i, and D_i are always located out of the polygonal obstacle *j*, and 2) each obstacle vertex is always located out of the vehicle body *i*. Thus the collision-avoidance constraints are written as

$$\mathrm{OutsidePolygon}(p, \mathrm{V}_{j1}, \ldots, \mathrm{V}_{j\mathrm{N}_j^{\mathrm{ver}}}), \forall p \in \{A_i(t), \ldots, D_i(t)\}, \forall j = 1, \ldots, \mathrm{N_{OBS}}, \forall i = 0, \ldots, \mathrm{N_V}, \forall t \in [0, T], \tag{10a}$$

and

$$\mathrm{OutsidePolygon}(q, A_i(t), \ldots, D_i(t)), \forall q \in \{\mathrm{V}_{j1}, \ldots, \mathrm{V}_{j\mathrm{N}_j^{\mathrm{ver}}}\}, \forall j = 1, \ldots, \mathrm{N_{OBS}}, \forall i = 0, \ldots, \mathrm{N_V}, \forall t \in [0, T]. \tag{10b}$$

$\mathrm{OutsidePolygon}(P, V_1 \sim V_{\mathrm{N}})$ contains the inequalities for the condition that point P is located outside convex polygon $V_1 \sim V_{\mathrm{N}}$. A triangle-area criterion is introduced in [28] to realize $\mathrm{OutsidePolygon}()$, which is a unified way to model polygon-to-polygon constraints without explicit involvement in if-else conditions.

Collisions between different bodies of the vehicle should also be avoided. Eq. (7) ensures that any vehicle body i has no chance to collide with $i+1$, $i+2$, and $i+3$. Thus, the interior collisions are avoided via the following constraints:

$$\mathrm{OutsidePolygon}(p, A_j(t), \ldots, D_j(t)), \forall p \in \{A_i(t), \ldots, D_i(t)\},$$
$$\mathrm{OutsidePolygon}(q, A_i(t), \ldots, D_i(t)), \forall q \in \{A_j(t), \ldots, D_j(t)\},$$
$$\forall i = 0, \ldots, \mathrm{N_V} - 4, \forall j = i+4, \ldots, \mathrm{N_V}, \forall t \in [0, T]. \tag{11}$$

Eq. (11) can be safely discarded when 1) $\mathrm{N_V} < 4$, or 2) vehicle body i can hardly reach vehicle body $i+4$. The second condition is true if the vehicle traverses in a tiny tunnel. Thus this preliminary study does not consider (11).

D. Cost Function

A tractor-trailer vehicle should reach the goal point with minimal time, thus the cost function is set to *T*. To conclude, a generic tractor-trailer vehicle maneuver planning task is described as the following OCP:

$$\begin{aligned} &\min T, \\ &\text{s.t. Kinematic constraints (1),(2),(4),(5),(6),(7);} \\ &\text{Two - point boundary constraints (8),(9);} \\ &\text{Collision - avoidance constraints (10).} \end{aligned} \tag{12}$$

The unknowns in (12) include T, $a_0(t)$, $\omega_0(t)$, $\phi_0(t)$, $x_i(t)$, $y_i(t)$, $v_0(t)$, and $\theta_i(t) (i = 0, \ldots, \mathrm{N_V})$.

III. Optimization-Based Maneuver Planner

A. Overall Framework

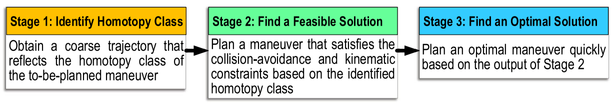
Stage 1: Identify Homotopy Class
Obtain a coarse trajectory that reflects the homotopy class of the to-be-planned maneuver
Stage 2: Find a Feasible Solution
Plan a maneuver that satisfies the collision-avoidance and kinematic constraints based on the identified homotopy class
Stage 3: Find an Optimal Solution
Plan an optimal maneuver quickly based on the output of Stage 2

Fig. 2. Overall framework of the proposed maneuver planning method.

The entire maneuver planning method comprises three stages (Fig. 2). A reference trajectory, which represents the homotopy class, is identified at Stage 1. A feasible or near-feasible maneuver based on the reference trajectory is identified at Stage 2. Finally, an optimal maneuver is planned at Stage 3.

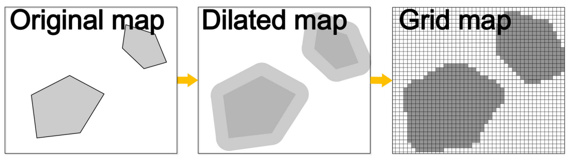

Fig. 3. Schematics on the grid map creation.

B. Stage 1: Identify the Homotopy Class

The tractor-trailer vehicle traverses in a tunnel-like scenario, thus there is only one homotopy class. Through generating a coarse trajectory at Stage 1 via a 2-dim A$*$ search in a grid map, the homotopy class can be explicitly specified.

1) Preparation for a Grid Map: A circumcircle Cir_i is deployed to cover each vehicle body $i(i = 0, \ldots, \mathrm{N_V})$. The radius of Cir_i,denoted as r_i,can be identified easily with elementary geometrics knowledge. Let us define $r = \max\{r_0, \ldots, r_{\mathrm{Nv}}\}$and dilate each obstacle by r to build a dilated map. The dilated map is thereafter sampled into mesh grids, all of which form a grid map. Fig. 3 schematically explains the operations to build the grid map.

2) Generation of Coarse Paths: Since a tractor-trailer vehicle consists of ($\mathrm{N_V}$+1) rigid bodies, we plan a coarse path for each body independently. For vehicle body $i(i = 0, \ldots, \mathrm{N_V})$, the initial and terminal locations of the center, denoted as (xc_i, yc_i), can be easily identified as per (4), (8), and (9). A$*$ algorithm is deployed to search for a sequence of nodes connecting the initial and terminal locations in the grid map. Although being kinematically infeasible, the resultant path is *definitely* collision-free in the original map because the path is searched in a map dilated by r, which is larger than or equal to r_i. The output of Stage 1 is a combination of ($\mathrm{N_V}$+1) coarse paths.

C. Stage 2: Find a Feasible Solution

The purpose of Stage 2 is to quickly plan a *feasible* maneuver based on the identified homotopy class at Stage 1. The feasibility refers to the satisfaction of vehicle kinematics, collision-avoidance constraints, and boundary constraints.

Nominally, the concerned maneuver planning scheme is completed by solving OCP (12) numerically, but that would be difficult due to the complex constraints (10). Instead of solving (12) directly, Stage 2 focuses on simplifying (10) because it is the main source of complexity in (12). Most of the constraints in (10) are redundant because a vehicle does not simultaneously have chances to collide with *every* obstacle at *every* moment. The principle to simplify (10) are presented as follows.

1) Generation of Reference Trajectories: A preliminary step is to covert the coarse paths into trajectories. Each of the ($\mathrm{N_V}$+1) paths derived at Stage 1 becomes a trajectory if a time-optimal velocity profile is attached to it. This is achieved by solving a 1-dim OCP via Pontryagin's maximum principle. Each resultant trajectory is resampled evenly in time as $(N_{FE} + 1)$ elements $traj = \{\sigma_0, \sigma_1, \ldots, \sigma_{N_{FE}}\}$, wherein each element σ_i records the locations of the tractor and $\mathrm{N_V}$ trailers $\sigma_i.x_0, \sigma_i.y_0, \ldots, \sigma_i.x_{\mathrm{Nv}}, \sigma_i.y_{\mathrm{Nv}}$, as well as a time stamp $\sigma_i.t$.

2) Constructions of Safe Travel Corridors: A safe travel corridor (STC) concept has been introduced in [21] to simplify the collision-avoidance constraints in a motion planning problem. A spatio-temporal corridor, which naturally separates a rigid-body vehicle from the surrounding obstacles, can be realized with a reference trajectory. The STC idea can be extended to handle the concerned multi-body vehicle case in this work if ($\mathrm{N_V}$ +1) spatio-temporal corridors are constructed. The concrete principle to construct a corridor based on a reference trajectory is found in [21].

The nominal collision-avoidance constraints (10) are then replaced by within-corridor constraints, which are tractably scaled and linear. Denoting the geometric center of vehicle body *k* as $P_k^{\mathrm{C}} = (xc_k, yc_k), k = 0, \ldots, \mathrm{N_V}$, the within-corridor constraints are then written as box constraints:

$$\begin{aligned}
&\mathrm{xc}_{\min}^{k,i} \le xc_k(t) \le \mathrm{xc}_{\max}^{k,i},\\
&\mathrm{yc}_{\min}^{k,i} \le yc_k(t) \le \mathrm{yc}_{\max}^{k,i},\\
&t \in (T/\mathrm{N_{FE}} \cdot (i-1), T/\mathrm{N_{FE}} \cdot i],\\
&i = 1, \ldots, \mathrm{N_{FE}},\ k = 0, \ldots, \mathrm{N_V}. \qquad (13)
\end{aligned}$$

The coordinate of each geometric center is defined as

$$\begin{aligned}
xc_k(t) &= x_k(t) + \frac{1}{2}(\mathrm{L}_{\mathrm{N}k} - \mathrm{L}_{\mathrm{M}k}) \cdot cos\theta_k(t),\\
yc_k(t) &= y_k(t) + \frac{1}{2}(\mathrm{L}_{\mathrm{N}k} - \mathrm{L}_{\mathrm{M}k}) \cdot \sin\theta_k(t), t \in [0, T]. \quad (14)
\end{aligned}$$

Eq. (13) represents the within-corridor constraints, and (14) shows the relationship between $(xc_k, yc_k)$and (x_k, y_k). The newly added algebraic equalities (14) can be taken as kinematic constraints. Notably, the scale of (13) or (14) is unrelated to $\mathrm{N_{OBS}}$, which means the constraint dimensionality is fully irrelevant to the environmental complexity. The corridor *k*, comprising $\mathrm{N_{FE}}$ axis-aligned local boxes, defines the local region for the *k*th vehicle body to stay in. Herein, a box is said to be "axis-aligned" if its edges are either parallel to the *x* or *y* axle. The location of the *i*th local box in corridor *k* is described by the four boundary values: $\mathrm{xc}_{\min}^{k,i}$, $\mathrm{xc}_{\max}^{k,i}$, $\mathrm{yc}_{\min}^{k,i}$, and $\mathrm{yc}_{\max}^{k,i}$. The aforementioned coarse trajectory $traj$ is used to identify the local boxes in corridor *k*.

The following new OCP is formulated by replacing (10) with (13) and (14):

$$\begin{aligned}
&\min T,\\
&\text{s.t. Kinematic constraints (1),(2),(4),(5),(6),(7),(14);}\\
&\text{Two - point boundary constraints (8),(9);}\\
&\text{Within - corridor constraints (13).} \qquad (15)
\end{aligned}$$

3) Formulation of Intermediate OCPs: Broadly speaking, maneuver planning is about resolving the underlying conflicts between the kinematics-related and environment-related restrictions. In our concerned OCP (15), the within-corridor constraints are in question because the corridors are constructed as per low-quality paths derived at Stage 1, thereby easily leaving out some free space necessary for kinematic feasibility. Thus, solving (15) numerically does not always work.

Instead of solving (15) directly, a lightweight iterative optimization strategy (LIOS) is proposed to address the aforementioned issue. LIOS builds an iterative framework, wherein an intermediate OCP is solved in each iteration. The derived optimum in one iteration serves as the reference trajectories to reconstruct the spatio-temporal corridors for solving an updated OCP in the next iteration, hoping to further reduce the solution infeasibility then. Eq. (16), a simplified version of (15) is defined as the aforementioned intermediate OCP. It is formed by softening the nonlinear kinematic constraints (1), (4), (5), (6), and (13) as

Algorithm 1: LIOS-Based Solution Feasibility Recovery Method.

```
Function LIOS (map,task,path)
 1.  $traj \leftarrow \mathrm{ConvertToTrajectory}(path)$;
 2.  $iter \leftarrow 0, \zeta \leftarrow +\infty$;
 3.  while $\zeta \geq \varepsilon_{\mathrm{tol}}$ do
 4.    $iter \leftarrow iter + 1$;
 5.    if $iter > \mathrm{iter}_{\max}$ then
 6.      break;
 7.    end if
 8.    $\Gamma \leftarrow \mathrm{GenerateCorridors}(map, traj)$;
 9.    $OCP_{\mathrm{INT}} \leftarrow \mathrm{GenerateOCP}(task, \Gamma)$;
10.    $[is\_failed, traj] \leftarrow \mathrm{SolveOCP}(OCP_{\mathrm{INT}}, traj)$;
11.    $\zeta \leftarrow \mathrm{MeasureInfeasibility}(traj)$;
12.  end while
13.  return with $traj$.
```

quadratic external penalty costs:

$$\begin{aligned} &\min T + \mathrm{w}_{\mathrm{penalty}} \cdot \zeta, \\ &\text{s.t. Kinematic constraints (2),(7);} \\ &\text{Two - point boundary constraints (8),(9);} \\ &\text{Within - corridor constraints (13).} \end{aligned} \tag{16}$$

Herein, ζ denotes the sum of the penalty costs related to (1), (4), (5), (6), and (13). $\mathrm{w}_{\mathrm{penalty}}$ is the weighting parameter. For the sake of brevity, the full form of ζ is not provided. Instead, the subtle penalty function related to (1), which is denoted as $J_{\mathrm{penalty}(1)}$, is given as an example:

$$\begin{aligned} J_{\mathrm{penalty}(1)} = &\int_{\tau=0}^{T} \left\| x'_0(\tau) - v_0(\tau)\cos\theta_0(\tau) \right\|^2 \mathrm{d}\tau \\ &+ \int_{\tau=0}^{T} \left\| y'_0(\tau) - v_0(\tau)\sin\theta_0(\tau) \right\|^2 \mathrm{d}\tau \\ &+ \int_{\tau=0}^{T} \left\| v'_0(\tau) - a_0(\tau) \right\|^2 \mathrm{d}\tau \\ &+ \int_{\tau=0}^{T} \left\| \phi'_0(\tau) - \omega_0(\tau) \right\|^2 \mathrm{d}\tau \\ &+ \int_{\tau=0}^{T} \left\| \theta'_0(\tau) - \frac{v_0(\tau)\tan\phi_0(\tau)}{\mathrm{L_{W0}}} \right\|^2 \mathrm{d}\tau. \end{aligned} \tag{17}$$

According to its definition, ζ is nonnegative; a full satisfaction of the softened nonlinear equality constraints is achieved only when $\zeta = 0$.

4) Overall Principle At Stage 2: The full pseudo-codes of Stage 2 are summarized as follows.

The inputs of LIOS() include *map*, *task*, and *path*. *map* presents the environmental setup, that is, the geometric shape and vertex location of each obstacle, as well as the scenario boundaries. *task* presents the planning scheme, that is, the initial and terminal configurations of the ego tractor-trailer vehicle. *path* denotes the coarse paths obtained at Stage 1. The output of LIOS() at Stage 2 is a series of trajectories sampled at ($\mathrm{N_{FE}}$ + 1) time instances $traj = \{\sigma_0, \sigma_1, \ldots, \sigma_{N_{FE}}\}$. In line 3 of Algorithm 1, $\varepsilon_{\mathrm{tol}} > 0$ is a user-specified convergence acceptance threshold. $\mathrm{iter}_{\max}$ in line 5 denotes the maximum allowable iteration number. The softened nonlinear constraints are regarded as satisfied if $\zeta < \varepsilon_{\mathrm{tol}}$. GenerateCorridors() identifies the local boxes in the ($\mathrm{N_V}$+1) corridors [21]. GenerateOCP() formulates an intermediate OCP in the form of (16). SolveOCP() solves an OCP numerically. *is_failed* is a Boolean output that reflects whether the OCP solution process succeeds, and *traj* records the optimum if available. MeasureInfeasibility(*traj*) calculates the infeasibility degree ζ of a given solution vector *traj*.

Remark 2.1 Compared with (15), each intermediate OCP (16) is always feasible because it no longer contains conflicting constraints. Moreover, (16) can be rapidly solved because all of its constraints are box constraints, i.e., the simplest type of linear constraints.

Remark 2.2 Alg. 1 runs fast because it has an upper bound on its iteration rounds. If a kinematically feasible is not found at Stage 2, we still regard the output of Alg. 1 as a near-feasible solution, and then pass it on to Stage 3 for further reducing the kinematic infeasibility.

D. Stage 3: Find an Optimal Solution

Stage 3 aims to find an optimal solution to the original OCP problem through a feasible or near-feasible maneuver *traj* derived at Stage 2. An intuitive idea is to use *traj* as the initial guess to warm-start the solution process of (12). However, this idea would be time-consuming because (12) contains intractably scaled collision-avoidance constraints. Instead of solving (12) directly, Stage 3 builds a trust-region-based iterative framework similar to LIOS().

1) Formulation of Intermediate OCPs: Stage 3 consists of an iterative framework wherein intermediate OCPs are solved and the collision-avoidance constraints are adaptively activated only when they are considered as necessary. In each intermediate OCP, spatio-temporal trust-region restrictions are imposed on the configurations of the tractor and trailers. In this way, the ego vehicle needs not consider the collision risks outside the trust regions. The optimum of one iteration serves as the reference trajectories to update the trust regions for the next iteration. The iteration continues until the cost function value converges.

With a given $traj = \{\sigma_0, \sigma_1, \ldots, \sigma_{N_{FE}}\}$,the following trust-region-based constraints are imposed on the configuration of each vehicle body throughout [0, *T*]:

$$\begin{aligned} &|x_k(t) - \sigma_i.x_k| \leq \Delta\mathrm{s}, |y_k(t) - \sigma_i.y_k| \leq \Delta\mathrm{s}, \\ &\quad \times |\theta_k(t) - \sigma_i.\theta_k| \leq \Delta\mathrm{a}, \\ &t \in (T/\mathrm{N_{FE}} \cdot (i-1), T/\mathrm{N_{FE}} \cdot i], i = 1, \ldots, \mathrm{N_{FE}}, \\ &\quad k = 0, \ldots, \mathrm{N_V}. \end{aligned} \tag{18}$$

Herein, $\Delta\mathrm{s}, \Delta\mathrm{a} > 0$ are user-specified parameters that decide the spatial range of the trust regions in the $x{-}y{-}\theta$ space. For vehicle body *k* during the *i*th time interval $t \in (T/\mathrm{N_{FE}} \cdot (i-1), T/\mathrm{N_{FE}} \cdot i]$, whether vehicle body *k* has chances to collide with each of the static obstacles is evaluated by the possible footprints of the vehicle body *k* which are enumeratively sampled within the trust regions. If there is no collision chance between vehicle body *k* and obstacle *j* at the *i*th time interval, then the corresponding collision-avoidance constraints would be removed from (10). The following intermediate OCP is formulated considering the aforementioned analysis:

$$\begin{aligned} &\min T, \\ &\text{s.t. Kinematic constraints (1),(2),(4),(5),(6),(7);} \\ &\text{Two - point boundary constraints (8),(9);} \\ &\text{Trust - region constraints (18);} \\ &\text{Reduced collision - avoidance constraints.} \end{aligned} \tag{19}$$

Algorithm 2: Trust-region-based iterative optimization method

```
Function TRMO($map, task, traj$)
1.  $iter \leftarrow 0$;
2.  $T_{\text{previous}} \leftarrow +\infty$;
3.  $T_{\text{current}} \leftarrow traj.\sigma_{N_{FE}}.t$;
4.  while $T_{\text{previous}} - T_{\text{current}} \geq \text{c}_{\text{tol}}$ , do
5.      $iter \leftarrow iter + 1$;
6.      if $iter > \text{iter}_{\max 2}$ , then
7.          break;
8.      end if
9.      $OCP_{\text{INT}} \leftarrow$ GenerateOCP2($task, map, traj$);
10.     $[is\_failed, traj] \leftarrow$ SolveOCP($OCP_{\text{INT}}, traj$);
11.     if $is\_failed$ , then
12.         return with $traj \leftarrow \emptyset$;
13.     end if
14.     $T_{\text{previous}} \leftarrow T_{\text{current}}$;
15.     $T_{\text{current}} \leftarrow traj.\sigma_{N_{FE}}.t$;
16. end while
17. return with traj.
```

2) Overall Principle At Stage 3: The full pseudo-codes of the trust-region-based maneuver optimization (TRMO) framework at Stage 3 are summarized as follows.

Similar to LIOS(), the inputs of TRMO() include *map*, *task*, and *traj*. The output of TRMO() is a series of trajectories sampled at (N_{FE}+1) time instances $traj = \{\sigma_0, \sigma_1, \ldots, \sigma_{N_{FE}}\}$. $\text{c}_{\text{tol}} > 0$ is a threshold parameter to accept the convergence. $\text{iter}_{\max 2}$ denotes the maximum allowable iteration number. GenerateOCP2() formulates an intermediate OCP in the form of (19). If the solution of any intermediate OCP fails (line 11), then the entire Algorithm 2 exits immediately.

Remark 3.1 In Alg. 2, the only possible chance to activate line 11 is when $iter = 1$. This is because a valid solution derived in $iter \geq 1$ is definitely a feasible starting point in the next iteration $iter + 1$. If the first iteration round really fails, possible reasons include 1) the solution obtained at Stage 2 is still far from being feasible so that no feasible solution ever exit in our imposed trust region around it, and 2) the original maneuver planning task is illegal.

Remark 3.2 Line 6 of Alg. 2 ensures that the optimization process does not take excessively long. Empirically, an optimized rather than optimal solution is also acceptable if a strict limitation on the runtime is also important. This design enables TRMO to have a sense of *anytime*.

E. Overall Maneuver Planning Method

As a summary of this section, the proposed tractor-trailer vehicle maneuver planning method is described as follows.

Algorithm 3: Tractor-Trailer Vehicle Maneuver Planning Method.

```
Function TRTT (map,task)
1.  $path \leftarrow$
    GenerateReferencePathViaAstar($map, task$);
2.  $traj \leftarrow$ LIOS($map, task, path$);
3.  $traj \leftarrow$ TRMO($map, task, traj$);
4.  if $traj \neq \emptyset$ , then
5.      return with traj;
6.  else
7.      return with a failure;
8.  end if
```

TABLE I
CRITICAL PARAMETRIC SETTINGS

Parameter	Setting
L_{N0}, L_{M0}, L_{W0}, L_{B0}, buffer	0.25 m, 0.25 m, 1.50 m, 2.00 m, 0.10 rad
$\text{L}_{\text{N}i}$, $\text{L}_{\text{M}i}$, $\text{L}_{\text{B}i}$, $\text{L}_{\text{H}(i-1)}$ (i = 1,…,N_{V})	1.00 m, 1.00 m, 2.00 m, 0 m
$\text{a}_{\max}$, $\text{v}_{\max}$, $\Phi_{\max}$, $\Omega_{\max}$	0.25 m/s^2, 2.50 m/s, 0.7 rad, 0.50 rad/s
N_{V}, N_{FE}, $\text{w}_{\text{penalty}}$, ε_{tol} , $\text{iter}_{\max}$	3, 100, 10^4, 10^{-3}, 5
c_{tol}, Δs , Δa , $\text{iter}_{\max 2}$	1.0 s, 3.00 m, π/4 rad, 10

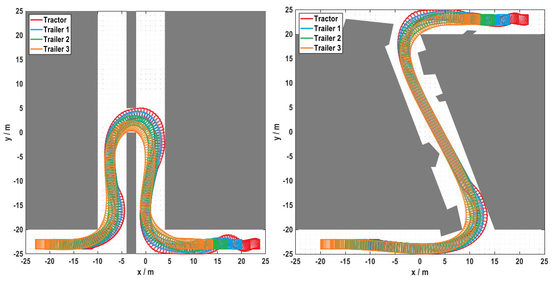

Fig. 4. Simulation results of Cases 1 and 2. Visibility of this figure can be improved if one zooms in the figure.

TABLE II
COMPARATIVE SIMULATION RESULTS I

	Case 1		Case 2	
	Cost	CPU time / s	Cost	CPU time / s
This work	44.61	4.44	47.12	6.38
PCOC	44.61	106.49	47.12	100.33
EHA	59.75	16.53	76.33	27.25
CSOC	53.05	26.59	67.04	43.96

IV. SIMULATION RESULTS AND DISCUSSIONS

Simulations are conducted in Matlab and executed on an i5-4460T CPU with 8 GB RAM that runs at 1.90×2 GHz. Regarding the function SolveOCP() defined in Section III, the first-order explicit Runge-Kutta method is adopted to form the NLP problem, which is solved by a primal-dual interior-point solver named IPOPT [29] with the linear solver set to MA97 [30] in AMPL [31]. Basic parametric settings are listed in Table I.

The proposed planner is tested in two cases. Simulation results are shown at www.bilibili.com/BV17o4y1D79h. Fig. 4 demonstrates the footprints in association with the obtained optimal maneuvers.

A. Comparisons With Existing Maneuver Planners

Comparisons with existing planners are made to evaluate the performance of the proposed maneuver planner. The extended hybrid A$*$ (EHA) algorithm is a typical sampling-and-search-based planner that samples in the control space for a tractor-trailer vehicle [10]. We additionally introduce a global route to attract the heuristic search along the tunnel central line to accelerate the search process in the tunnel. To make the comparisons fair, the path derived by EHA is attached to a minimal-optimal velocity to become a trajectory. The progressively constrained optimal control (PCOC) approach [10] is an optimization-based planner that refines the trajectory obtained by EHA. Tightly combining sampling-and-search and optimal control (CSOC) [20] refers to getting a coarse trajectory via EHA and solving local trajectory planning problems along segmented pieces via numerical OCP. The results are reported in Table II.

EHA is deployed to represent the prevalent sampling-and-search-based planners, such as the state-lattice planner [11] and

TABLE III
DEFINITIONS OF COMPARATIVE ALGORITHM VARIANTS

ID	Description
Alg. 4.1	Use the output of Stage 1 to warm-start the numerical solution process of OCP (12) directly
Alg. 4.2	Same with the proposed planner, except that Stage 3 solves OCP (12) for once instead of deploying an iterative framework
Alg. 4.3	Same with the proposed planner, except that Stage 2 is removed, that is, the output of Stage 1 is directly fed to Stage 3 for iterative optimization
Alg. 4.4	Same with the proposed planner, except that Stage 2 solves an OCP in the form of (16) for once rather than deploying LIOS
Alg. 4.5	Same with Alg. 4.1, except that Stage 1 is replaced by EHA
Alg. 4.6	Same with the proposed planner, except that Stage 1 is replaced by EHA

TABLE IV
COMPARATIVE SIMULATION RESULTS II

	Case 1		Case 2	
	Cost	CPU time / s	Cost	CPU time / s
This work	44.61	4.44	47.12	6.38
Alg. 4.1	Failed	15.27	Failed	24.07
Alg. 4.2	44.61	10.47	47.12	21.85
Alg. 4.3	Failed	6.96	Failed	5.65
Alg. 4.4	Failed	5.62	Failed	8.27
Alg. 4.5	44.61	35.39	47.12	74.47
Alg. 4.6	44.61	20.86	47.12	33.34

closed-loop RRT [13]. A sampling-and-search-based planner is typically known to run fast, but it requires many iterations to find a solution in a tiny tunnel-like environment, especially when the trailer kinematics complicates the problem. Limited by the sampling resolution issue, EHA finds a feasible but non-optimal solution slowly. PCOC also runs slowly because it 1) relies on precise sampling and search, and 2) attempts to solve the full-scale OCP (12). CSOC requires less runtime than PCOC. However, CSOC suffers from optimality loss because the segmented OCP solution process relies deeply on a sampled-and-searched trajectory, which fixes the two-point boundary constraints. By contrast, our proposed planner outperforms the others w.r.t. CPU runtime and optimality.

B. Comparisons With Variants of the Proposed Planner

The performance of each stage in our proposed planner deserves to be further evaluated. Six variants of the proposed planner are defined in Table III, and the results are listed in Table IV.

The inefficiency of Algorithm 4.1 indicates that a numerical-OCP method relies on high-quality initial guesses rather than poor ones. Compared with Algorithm 4.1, Algorithm 4.2 is effective because it has an extra stage (i.e., Stage 2) to identify a near-feasible initial guess. Algorithm 4.2 requires more runtime than our proposed method because Algorithm 4.2 is not able to remove the redundant constraints. In the two tested cases, our proposed method respectively discards 81% and 85% of all the collision-avoidance constraints in the first iteration of Stage 3. That is why our proposed planner runs faster than Algorithm 4.2 without a loss of optimality. Algorithm 4.3 is deployed to show that Stage 2 is a necessary module to refine a low-quality solution before its usage as an initial guess. Algorithm 4.4 differs from our proposed planner only in the intermediate OCP formulation at Stage 2. Iteratively solving (15) rather than (16) suffers from the risk of encountering an intermediate failure, which inevitably destroys the iteration framework. Algorithm 4.5 warm-starts the numerical solution process of the full-scale OCP (12) with a good initial guess derived by EHA, thus Algorithm 4.5 can find

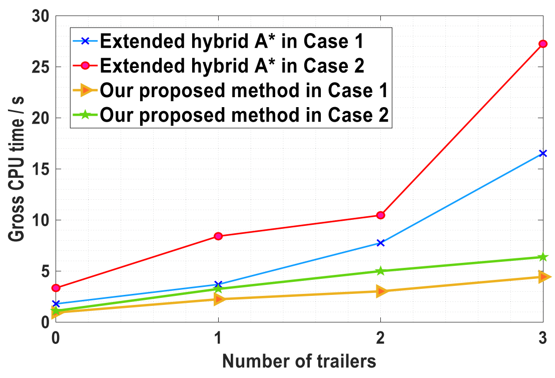

Fig. 5. Migration of runtime with $\mathrm{N_V}$. Our proposed optimization-based planner and a typical sampling-and-search-based planner EHA are tested.

optima while Algorithm 4.1 cannot. However, Algorithm 4.5 still runs slowly due to the large-scale OCP formulation it handles. Algorithm 4.6 uses EHA to replace Stage 1 of our proposed planner. Since EHA spends much runtime, Algorithm 4.6 is not fast. However, it deserves to note that the CPU runtime consumed at Stages 2 and 3 of Algorithm 4.6 *does* show a decline rate of 2% and 5% respectively than our method because of the high-quality coarse paths EHA provides at Algorithm 4.6's Stage 1. This indicates that promoting the solution quality of Stage 1 really works to save the subsequent optimization runtime although the improvement is rather minor. The result of Algorithm 4.6 confirms that planning without high-accuracy sampling and search, as our proposed maneuver planner does, is advantageous.

In addition to the six algorithm variants listed in Table III, Algorithm 4.7 is defined by replacing the A$*$ algorithm at Stage 1 with RRT$*$ [32]. Algorithm 4.7 is independently repeated for 1000 times to see the statistical significance because RRT$*$ is not deterministic. As it turns out, simulations on all of the repeated cases succeed. This indicates that: 1) our proposed planner supports different types of tools at its Stage 1, and 2) Stages 2 and 3 are really insensitive to the output quality of Stage 1.

C. Algorithm Tractability and Parametric Robustness

This subsection is focused on the tractability and robustness of our proposed planner. The first round of tests is done by gradually resetting $\mathrm{N_V}$ (i.e., the trailer number) from 3 to 0 and re-conducting the simulations accordingly. In Fig. 5, the runtime spent by the proposed optimization-based planner does not increase too fast with $\mathrm{N_V}$ whereas the sampling-and-search-based EHA performs conversely. This result shows that an optimization-based method is promising to tackle complex planning problems.

The second round of tests is done by resetting some critical parameters in the proposed planner. Only $\mathrm{w_{penalty}}$ and $\Delta \mathrm{s}$ are considered due to page limitation, and the results are depicted in Fig. 6. Various settings of $\mathrm{w_{penalty}}$ do not substantially alter the runtime. Thus, the proposed planner is insensitive to this parameter at Stage 2. Setting $\Delta \mathrm{s}$ too small or large generally renders additional runtime. This is because 1) a small $\Delta \mathrm{s}$ indicates small trust regions and additional iterations, and 2) a large $\Delta \mathrm{s}$ renders more collision-avoidance constraints to handle at Stage 3. Compared to the proposed planner, PCOC and EHA are quite sensitive to their critical parameters such as the sampling resolution and obstacle expansion step length.

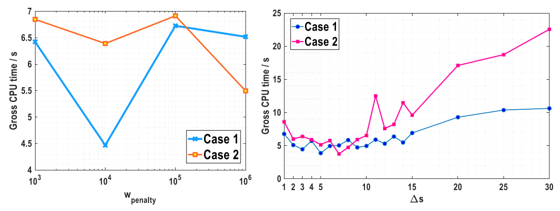

Fig. 6. Migration of runtime with different settings of $\mathrm{w_{penalty}}$ orΔs.

V. Conclusion

This letter has proposed an optimization-based multi-stage maneuver planner for a tractor-trailer vehicle in a curvy and tiny tunnel. Our planner has the merits of being fast, optimal, lightweight, robust, and insensitive to parametric settings.

A sampling-and-search-based planner is commonly the first choice in dealing with a maneuver planning problem in the community of autonomous driving. But this work aims to advocate the importance of optimization-based methods, whose potential to handle harsh high-dimensional problems deserves to be better exploited. If well designed, an optimization-based planner can do more than just a trajectory smoother which is fed to the sampled-and-searched initial guess.

In using a numerical OCP method, violations of collision-avoidance constraints between adjacent collocation points deserve to be resolved in a thorough and theoretical way, which is our work in the near future.

References

[1] B. Li, Y. Zhang, T. Acarman, Q. Kong, and Y. Zhang, "Trajectory planning for a tractor with multiple trailers in extremely narrow environments: A unified approach," in *Proc. IEEE Int. Conf. Robot. Automat.*, Aug. 2019, pp. 8557–8562.

[2] P. Ritzen, E. Roebroek, N. van de Wouw, Z. Jiang, and H. Nijmeijer, "Trailer steering control of a tractor–trailer robot," *IEEE Trans. Control Syst. Technol.*, vol. 24, no. 4, pp. 1240–1252, Jul. 2016.

[3] B. Li and Z. Shao, "Precise trajectory optimization for articulated wheeled vehicles in cluttered environments," *Adv. Eng. Softw.*, vol. 92, pp. 40–47, 2016.

[4] J. Liu, X. Dong, J. Wang, C. Lu, X. Zhao, and X. Wang, "A novel EPT autonomous motion control framework for an off-axle hitching tractor-trailer system with drawbar," *IEEE Trans. Intell. Veh.*, vol. 6, no. 2, pp. 376–385, Jun. 2021.

[5] M. Yue, X. Hou, X. Zhao, and X. Wu, "Robust tube-based model predictive control for lane change maneuver of tractor-trailer vehicles based on a polynomial trajectory," *IEEE Trans. Syst., Man, Cybern.: Syst.*, vol. 50, no. 12, pp. 5180–5188, Dec. 2020.

[6] H. Zhao, S. Zhou, W. Chen, Z. Miao, and Y. -H. Liu, "Modeling and motion control of industrial tractor–trailers vehicles using force compensation," *IEEE/ASME Trans. Mechatronics*, vol. 26, no. 2, pp. 645–656, Apr. 2021.

[7] B. Li, K. Wang, and Z. Shao, "Time-optimal trajectory planning for tractor-trailer vehicles via simultaneous dynamic optimization," In *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Dec. 2015, pp. 3844–3849.

[8] M. Hejase, J. Jing, J. M. Maroli, Y. Bin Salamah, L. Fiorentini, and Ü. Özgüner,, "Constrained backward path tracking control using a plug-in jackknife prevention system for autonomous tractor-trailers," In *Proc. 21st Int. Conf. Intell. Transp. Syst.*, Dec. 2018, pp. 2012–2017.

[9] J. Jing, J. M. Maroli, Y. B. Salamah, M. Hejase, L. Fiorentini, and Ü. Özgüner, "Control method designs and comparisons for tractor-trailer vehicle backward path tracking," In *Proc. Amer. Control Conf.*, 2019, pp. 5531–5537.

[10] B. Li, T. Acarman, Y. Zhang, C. Yaman, and Q. Kong, "Tractor-trailer vehicle trajectory planning in narrow environments with a progressively constrained optimal control approach," *IEEE Trans. Intell. Veh.*, vol. 5, no. 3, pp. 414–425, Sep. 2020.

[11] O. Ljungqvist, N. Evestedt, M. Cirillo, D. Axehill, and O. Holmer, "Lattice-based motion planning for a general 2-trailer system," In *Proc. IEEE Intell. Veh. Symp.*, Jul. 2017, pp. 819–824.

[12] J. W. Choi and K. Huhtala, "Constrained global path optimization for articulated steering vehicles," *IEEE Trans. Veh. Technol.*, vol. 65, no. 4, pp. 1868–1879, Apr. 2016.

[13] N. Evestedt, O. Ljungqvist, and D. Axehill, "Motion planning for a reversing general 2-trailer configuration using closed-loop RRT," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Dec. 2016, pp. 3690–3697.

[14] A. J. Rimmer and D. Cebon, "Planning collision-free trajectories for reversing multiply-articulated vehicles," *IEEE Trans. Intell. Transp. Syst.*, vol. 17, no. 7, pp. 1998–2007, Jul. 2016.

[15] M. Cirillo, "From videogames to autonomous trucks: A new algorithm for lattice-based motion planning," In *Proc. 2017 IEEE Intell. Veh. Symp.*, Jul. 2017, pp. 148–153.

[16] S. Beyersdorfer and S. Wagner, "Novel model-based path planning for multi-axle steered heavy load vehicles," In *Proc. 16th Int. IEEE Conf. Intell. Transp. Syst.*, Oct. 2013, pp. 424–429.

[17] P. Ferbach, "A method of progressive constraints for nonholonomic motion planning," *IEEE Trans. Robot. Automat.*, vol. 14, no.no. 1, pp. 172–179, Feb. 1998.

[18] A. Mohamed, J. Ren, H. Lang, and M. El-Gindy, "Optimal collision- free path planning for an autonomous articulated vehicle with two trailers," In *Proc. 2017 IEEE Int. Conf. Ind. Technol.*, Mar. 2017, pp. 860–865.

[19] J. V. Miró, A. S. White, and R. Gill, "On-line time-optimal algorithm for manipulator trajectory planning," in *Proc. 1997 Eur. Control Conf.*, Jul. 1997, pp. 2611–2616.

[20] K. Bergman, O. Ljungqvist, and D. Axehill, "Improved path planning by tightly combining lattice-based path planning and optimal control," *IEEE Trans. Intell. Veh.*, vol. 6, no. 1, pp. 57–66, Mar. 2021.

[21] B. Li *et al.*, "Optimization-based trajectory planning for autonomous parking with irregularly placed obstacles: A lightweight iterative framework," *IEEE Trans. Intell. Transp. Syst.*, early access, Sep. 2021.

[22] J. Ziegler, P. Bender, T. Dang, and C. Stiller, "Trajectory planning for Bertha—A local, continuous method," In *Proc. 2014 IEEE Intell. Veh. Symp. Proc.*, Jul. 2014, pp. 450–457.

[23] C. Sun, Q. Li, B. Li, and L. Li, "A successive linearization in feasible set algorithm for vehicle motion planning in unstructured and low-speed scenarios," *IEEE Trans. Intell. Transp. Syst.*, early access, Jan. 2021.

[24] H. Cen, B. Li, and T. Acarman, Y. Zhang, Y. Ouyang, and Y. Dong, "Optimization-based maneuver planning for a tractor-trailer vehicle in complex environments with safe travel corridors," In *Proc. 32nd IEEE Intell. Veh. Symp.*, Nov. 2021, pp. 974–979.

[25] H. Andreasson, J. Saarinen, M. Cirillo, T. Stoyanov, and A. J. Lilienthal, "Fast, continuous state path smoothing to improve navigation accuracy," in *Proc. 2015 IEEE Int. Conf. Robot. Automat.*, Jul. 2015, pp. 662–669.

[26] B. Li and Z. Shao, "An incremental strategy for tractor-trailer vehicle global trajectory optimization in the presence of obstacles," In *Proc. 2015 IEEE Int. Conf. Robot. Biomimetics*, Feb. 2015, pp. 1447–1452.

[27] M. Beglini, L. Lanari, and G. Oriolo, "Anti-jackknifing control of tractor-trailer vehicles via intrinsically stable MPC," In *Proc. 2020 IEEE Int. Conf. Robot. Automat.*, Sep. 2020, pp. 8806–8812.

[28] B. Li and Z. Shao, "A unified motion planning method for parking an autonomous vehicle in the presence of irregularly placed obstacles," *Knowl.-Based Syst.*, vol. 86, pp. 11–20, 2015.

[29] A. Wächter and L. Biegler, "On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming," *Math. Program.*, vol. 106, no. 1, pp. 25–57, 2006.

[30] HSL, "A collection of fortran codes for large scale scientific computation," [Online]. Available: http://www.hsl.rl.ac.uk/

[31] R. Fourer, D. M. Gay, and B. W. Kernighan, *AMPL: A Modeling Language for Mathematical Programming*. South San Francisco: The Scientific Press, 2003.

[32] Y. Kuwata, J. Teo, G. Fiore, S. Karaman, E. Frazzoli, and J. P. How, "Real-time motion planning with applications to autonomous urban driving," *IEEE Trans. Control Syst. Technol.*, vol. 17, no. 5, pp. 1105–1118, Jul. 2009.