Autonomous Driving on Curvy Roads Without Reliance on Frenet Frame: A Cartesian-Based Trajectory Planning Method

Bai Li, *Member, IEEE*, Yakun Ouyang, *Graduate Student Member, IEEE*, Li Li, *Fellow, IEEE*, and Youmin Zhang, *Senior Member, IEEE*

***Abstract*— Curvy roads are a particular type of urban road scenario, wherein the curvature of the road centerline changes drastically. This paper is focused on the trajectory planning task for autonomous driving on a curvy road. The prevalent on-road trajectory planners in the Frenet frame cannot impose accurate restrictions on the trajectory curvature, thus easily making the resultant trajectories beyond the ego vehicle's kinematic capability. Regarding planning in the Cartesian frame, selection-based methods suffer from the curse of dimensionality. By contrast, optimization-based methods in the Cartesian frame are more flexible to find optima in the continuous solution space, but the new challenges are how to tackle the intractable collision-avoidance constraints and nonconvex kinematic constraints. An iterative computation framework is proposed to accumulatively handle the complex constraints. Concretely, an intermediate problem is solved in each iteration, which contains linear and tractably scaled collision-avoidance constraints and softened kinematic constraints. Compared with the existing optimization-based planners, our proposal is less sensitive to the initial guess especially when it is not kinematically feasible. The efficiency of the proposed planner is validated by both simulations and real-world experiments. Source codes of this work are available at https://github.com/libai1943/CartesianPlanner.**

***Index Terms*— Trajectory planning, computational optimal control, nonlinear program, Frenet frame, autonomous vehicle.**

I. Introduction

AUTONOMOUS driving techniques are promising to promote travel safety, comfort, and mobility in an urban transportation system [1]–[3]. Typical on-board modules in an autonomous vehicle include perception, planning, and control.

Manuscript received June 28, 2021; revised November 26, 2021 and January 1, 2022; accepted January 19, 2022. This work was supported in part by the National Key Research and Development Program of China under Grant 2020AAA0108104; in part by the Fundamental Research Funds for the Central Universities under Grant 531118010509; in part by the National Natural Science Foundation of China under Grant 62103139, Grant 61873047, and Grant 61833013; in part by the Natural Science Foundation of Hunan Province, China, under Grant 2021JJ40114; and in part by the Natural Sciences and Engineering Research Council of Canada. The Associate Editor for this article was Z. Xiao. *(Corresponding authors: Li Li; Youmin Zhang.)*

Bai Li and Yakun Ouyang are with the College of Mechanical and Vehicle Engineering, Hunan University, Changsha 410082, China (e-mail: libai@zju.edu.cn; yakun@hnu.edu.cn).

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

Youmin Zhang is with the Department of Mechanical, Industrial and Aerospace Engineering, Concordia University, Montreal, QC H3G 1M8, Canada (e-mail: ymzhang@encs.concordia.ca).

Digital Object Identifier 10.1109/TITS.2022.3145389

(a) (b)

Fig. 1. Typical curvy road scenarios: (a) an urban curvy road; (b) a straight road complicated by roadside obstacles.

Specifically, the planning module is responsible for producing open-loop trajectories for low-level controllers to track. The trajectory planning scheme in the planning module is a direct reflection of the intelligence level of an autonomous vehicle. The existing trajectory planners are typically classified as the ones for on-road cruising and off-road parking. An on-road planner primarily cares about how to make the curvature of a trajectory limited and continuous while keeping the trajectory from collisions [4]. A parking-oriented planner focuses on how to steer a vehicle to a desired configuration via a few maneuvers [5]. Typically, on-road trajectories are smooth while parking trajectories include cusps. In most cases, on-road planners and parking planners are quite different. This study focuses on the trajectory planning task for on-road autonomous driving.

Curvy roads are a particular type of urban road scenario, wherein the curvature of the road centerline changes drastically (Fig. 1a). Even if the road is straight, the irregularly placed roadside obstacles make the road tiny, thus rendering curvy trajectories. Such scenarios are not rare on an urban road in developing countries (Fig. 1b).

As the curvature of the road trend is high on a curvy road, the trajectory should be carefully planned to ensure it fits the kinematic capability of the ego vehicle, otherwise the controller would not be able to track the open-loop trajectory. As we will explain later, although there have been many publications about on-road trajectory planning [1], [4], [6], few of them can handle curvy road scenarios well. This paper is about trajectory planning on a curvy road.

A. Related Works

An on-road trajectory planner is regarded as capable of handling curvy roads if it can guarantee that the trajectory

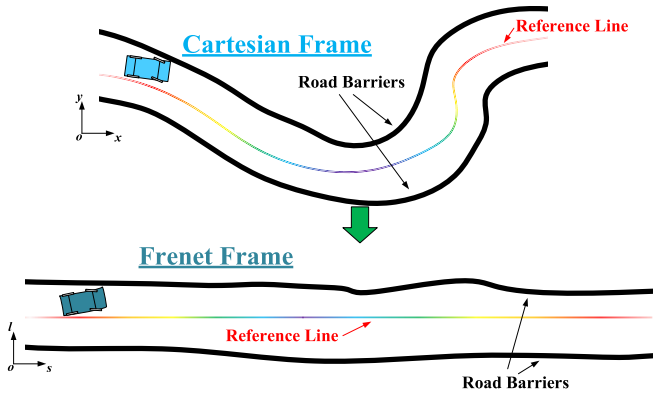


Fig. 2. Schematics on the conversion from Cartesian frame to Frenet frame.

curvature does not exceed the vehicle kinematic limits. Unfortunately, majority of the planners in this community are not qualified due to the usage of Frenet frame.

The Frenet frame, also known as the curvilinear frame, has been widely used to model an on-road trajectory planning task [7]. As depicted in Fig. 2, an irregularly shaped road in the real-world Cartesian frame is converted into a straight one in the Frenet frame. An obvious benefit of using the Frenet frame is that any road can be standardized as a straight tunnel with left and right bounds. In this way, the nonlinear collision-avoidance constraints in a trajectory planning problem are converted into linear within-tunnel constraints. Besides that, the originally coupled kinematic constraints are decoupled as independent polynomials in the longitudinal and lateral dimensions [8]. The aforementioned features enable a Frenet-based method to describe the trajectory planning scheme as a quadratic program (QP), which can be quickly solved [9]–[14].

Despite the aforementioned merits, the usage of Frenet frame has the following side effects: 1) disability to model the true kinematics of the ego vehicle, 2) ignorance of vehicle shape distortion, 3) failure to ensure trajectory continuity, and 4) disability to support multi-vehicle cooperative planning. The first side effect easily renders violations of trajectory curvature limits because a Frenet-based planner is not aware of the vehicle's actual kinematic capability. The situation becomes worse when the road is curvy. The rationale behind these arguments is elaborated in Appendix A. To summarize, the Frenet frame is not suitable for modeling trajectory planning problems when the road is curvy.

Since the Frenet-based planners are not suitable for autonomous driving on the curvy roads, one may alternatively consider modeling the planning task back in the Cartesian frame, wherein the vehicle kinematic or even dynamic capability can be adequately described. The Cartesian-based planners suitable for curvy roads are classified as selection- and optimization-based methods. A selection-based method is featured by checking the feasibility of each candidate trajectory in the Cartesian frame. Li *et al.* [8] sampled candidate trajectories in the Frenet frame and regularized them before evaluating their costs in the Cartesian frame. Kuwata *et al.* [15] adopted a virtual controller to track the trajectory primitives after they were sampled via rapidly exploring random tree (RRT) search; each tracking result was taken as a trajectory candidate for selection, which naturally satisfies the kinematic feasibility in the Cartesian frame. Ma *et al.* [16] post-processed the RRT primitives via model predictive control methods to enhance the kinematic feasibility of each candidate trajectory primitive in the Cartesian frame. A similar idea was proposed by Li *et al.* in [17]. Since the trajectory curvature in the real-world Cartesian frame can be precisely measured, the selection-based planners can deal with curvy roads. However, selecting among finite primitives makes a planner incomplete, especially when none of the primitives can satisfy the kinematic and collision-avoidance constraints on a curvy road.

In contrast with the selection-based methods that choose from finite candidates, an optimization-based planner can find an optimum among infinite candidates in the continuous solution space, thus being more flexible than a selection-based method [18], [19]. An optimization-based planner describes the concerned on-road trajectory planning task as an optimal control problem (OCP) and then solves it numerically via a gradient-based optimizer. Through modeling the vehicle kinematics as hard constraints in the Cartesian frame, the trajectories planned via an optimization-based method naturally satisfy the kinematic principle. Although an optimization-based planner has merits in the kinematic feasibility and solution flexibility, it faces two new challenges [20], which are introduced as follows.

The first challenge is, the collision-avoidance constraints in the Cartesian frame are far more complex than the ones in the Frenet frame because the drivable area on the curvy road is no longer a standardized tunnel with simply left and right bounds. Since the ego vehicle's contour is not smooth, the collision-avoidance constraints w.r.t. the road barriers and obstacles are nominally non-differentiable [21], which are beyond the capability of a gradient-based OCP solver. To address this issue, a common idea is to present the collision-avoidance constraints via within-corridor constraints after constructing a spatio-temporal safe corridor. Ziegler *et al.* [22] divided the ego vehicle body into multiple parts and constructed corridors for each of the parts, but the pseudo distance functions in the within-corridor constraints are still non-convex, thus rendering a large runtime as indicated by [21]. Liu *et al.* [23] proposed a convex feasible set method to make the within-corridor constraints linear, but the dimension of within-corridor constraints is not fixed, thereby making the OCP solution performance vary. To fix the scale of the within-corridor constraints, Ding *et al.* [24], Li and Zhang [25], and Manzinger *et al.* [14] constructed the corridor with a series of rectangles aligned to a reference trajectory. However, all the aforementioned corridor construction strategies inevitably leave out drivable area in part, thus rendering solution failures if the left-out region is necessary for kinematic feasibility. More importantly, the correctness of the within-corridor constraints relies too much on the quality of the reference trajectory. If the reference trajectory is far from being kinematically feasible, then the OCP easily becomes infeasible.

The second challenge is about the nonconvexity of the Cartesian-based kinematic constraints, which largely pull down the OCP solution speed. To address this issue, Zhang *et al.* [26] assumed that the curvature profile is a parameterized cubic spline, thereby simplifying the nominal

kinematic constraints. However, the usage of specified splines to describe the state/control profiles in an OCP would reduce the planning flexibility, thus rendering solution failures easily. Liniger *et al.* [27] built an iterative framework, wherein a QP problem with linearized kinematic constraints is repeatedly solved; the optimum derived in one iteration serves as the first-order Taylor expansion point in the next iteration; the collision-avoidance constraints are relaxed with slack variables while requiring to minimize the slack variables; the iteration continues until convergence. However, the efficiency of this method relies highly on good first-order Taylor expansion points, without which the intermediate QP problem becomes infeasible and thus terminates the entire iterative process.

To summarize, the state-of-the-art optimization-based planners in the Cartesian frame are still not perfect in dealing with a curvy road scenario.

B. Motivations and Contributions

This study aims to propose a fast, precise, and optimal trajectory planning method for autonomous driving on curvy roads. Since the Frenet-based planners cannot accurately describe the vehicle kinematics, we choose to develop a Cartesian-based planning method. More specifically, since the selection-based methods suffer from the curse of dimensionality, we choose to develop an optimization-based planner due to its potential to find an optimum in the continuous solution space.

The core contribution of this paper is the proposal of an iterative computation framework, whereby the two challenges an optimization-based planner encounters can be efficiently addressed. Compared with the existing optimization-based trajectory planning methods in the Cartesian frame, our proposal guarantees that the within-corridor constraints are linear, tractably scaled, insensitive to the reference trajectory, and have sufficient coverage w.r.t. the drivable area. Also, our proposal simplifies the nonconvex kinematic constraints without violating them or making the planner reliant upon the initial guess. In addition, our proposed method handles the two challenges in an integrated way rather than raising two individual strategies.

In the rest of this paper, Section II states the on-road trajectory planning problem. Section III proposes our trajectory planner. Simulation and experimental results are reported and discussed in Section IV, followed by the conclusions drawn in Section V.

II. Problem Statement

In this section, the on-road trajectory planning task is nominally formulated as an OCP, which consists of a cost function and constraints, the details of which are given in the next few subsections.

A. Overall Formulation

A standard OCP in the form of (1) is deployed to describe the concerned trajectory planning task.

$$\begin{aligned}
&\text{Minimize } J\left(\mathbf{x}(t), \mathbf{u}(t)\right),\\
&\text{s.t. } \dot{\mathbf{x}}(t) = f_{\text{kinematics}}\left(\mathbf{x}(t), \quad \mathbf{u}(t)\right),\\
&\quad \mathbf{x} \le \mathbf{x}(t) \le \bar{\mathbf{x}}, \quad \mathbf{u} \le \mathbf{u}(t) \le \bar{\mathbf{u}}, \ t \in [0, \mathrm{T}];\\
&\quad \mathbf{x}(0) = \mathbf{x}_{\text{init}}, \quad \mathbf{u}(0) = \mathbf{u}_{\text{init}}, \ \mathbf{x}(\mathrm{T}) = \mathbf{x}_{\text{goal}},\\
&\quad \mathbf{u}(\mathrm{T}) = \mathbf{u}_{\text{goal}}; \quad h_{\text{collision}}\left(\mathbf{x}(t)\right) \le 0, \ t \in [0, \mathrm{T}]. \quad (1)
\end{aligned}$$

$\mathbf{x}(t)$ denotes the state profiles in the Cartesian frame. Concretely, it stands for $[x(t), y(t), \theta(t), v(t), a(t), \phi(t)]$, where $(x(t), y(t))$ denotes the location of a reference point on the ego vehicle (the reference point is set to the rear-axle midpoint in this work), $\theta(t)$ denotes the orientation angle of the vehicle, $v(t)$ is the longitudinal velocity, $a(t)$ is the corresponding acceleration, and $\phi(t)$ represents the steering angle. $\mathbf{u}(t)$ represents the control profiles $[jerk(t), \omega(t)]$ in the Cartesian frame, wherein $jerk(t)$ is the derivative of $a(t)$, and $\omega(t)$ is the angular velocity of $\phi(t)$. $f_{\text{kinematics}} = 0$ gathers the equalities to describe the vehicle kinematics. $h_{\text{collision}} \le 0$ stands for the collision-avoidance constraints. $[\mathbf{x}, \bar{\mathbf{x}}, \mathbf{u}, \bar{\mathbf{u}}]$ forms the allowable bounds of the state/control profiles. T denotes the planning horizon.

B. Cost Function

The cost function $J\left(\mathbf{x}(t), \mathbf{u}(t)\right)$ is defined as

$$J = \int_{\tau=0}^{\mathrm{T}} \left(\|\mathbf{x}(\tau) - \mathbf{x}_{\text{ref}}(\tau)\|^2 + \mathrm{w_u} \cdot \|\mathbf{u}(\tau)\|^2 \right) \cdot \mathrm{d}\tau, \quad (2)$$

where $\mathrm{w_u} > 0$ is a weighting parameter.

$\|\mathbf{x}(\tau) - \mathbf{x}_{\text{ref}}(\tau)\|^2$ is used to encourage the state of the ego vehicle toward the nominal one $\mathbf{x}_{\text{ref}}$ at time τ. Herein, $\mathbf{x}_{\text{ref}}$ represents the nominal driving status along a reference trajectory. The term $\|\mathbf{x}(\tau) - \mathbf{x}_{\text{ref}}(\tau)\|^2$ in (2) is concretely written as

$$\begin{aligned}
&\|\mathbf{x}(\tau) - \mathbf{x}_{\text{ref}}(\tau)\|^2\\
&\quad = (x(\tau) - \mathrm{x}_{\text{ref}}(\tau))^2\\
&\qquad + \left(y(\tau) - \mathrm{y}_{\text{ref}}(\tau)\right)^2 + \mathrm{w}_{\mathrm{r}\theta} \cdot (\theta(\tau) - \theta_{\text{ref}}(\tau))^2, \quad (3)
\end{aligned}$$

where $\left(\mathrm{x}_{\text{ref}}(\tau), \mathrm{y}_{\text{ref}}(\tau), \theta_{\text{ref}}(\tau)\right)$ denotes the reference trajectory parameterized by τ, and $\mathrm{w}_{\mathrm{r}\theta} \ge 0$ is a weighting parameter. A reference trajectory refers to a coarse trajectory derived in the upstream decision-making module. In this work, a reference trajectory is derived by searching in an abstracted state space via dynamic programming (DP), the technical details of which are available in [28] and [29].

The second term $\|\mathbf{u}(\tau)\|^2$ is deployed to encourage the control profiles towards zero, thereby promoting passenger comfort, enhancing trajectory smoothness, and saving energy. $\|\mathbf{u}(\tau)\|^2$ is concretely written as

$$\|\mathbf{u}(\tau)\|^2 = jerk^2(\tau) + \mathrm{w_{rw}} \cdot \omega^2(\tau), \quad (4)$$

wherein $\mathrm{w_{rw}} \ge 0$ is a weighting parameter.

C. Constraints

The constraints in a trajectory planning task typically include the kinematic constraints, two-point boundary constraints, and collision-avoidance constraints.

1) Kinematic Constraints: The kinematic principle of the ego vehicle is reflected by $f_{\text{kinematics}} = 0$, $\mathbf{x} \le \mathbf{x}(t) \le \bar{\mathbf{x}}$, and $\mathbf{u} \le \mathbf{u}(t) \le \bar{\mathbf{u}}$ in (1). Their concrete forms are given as follows.

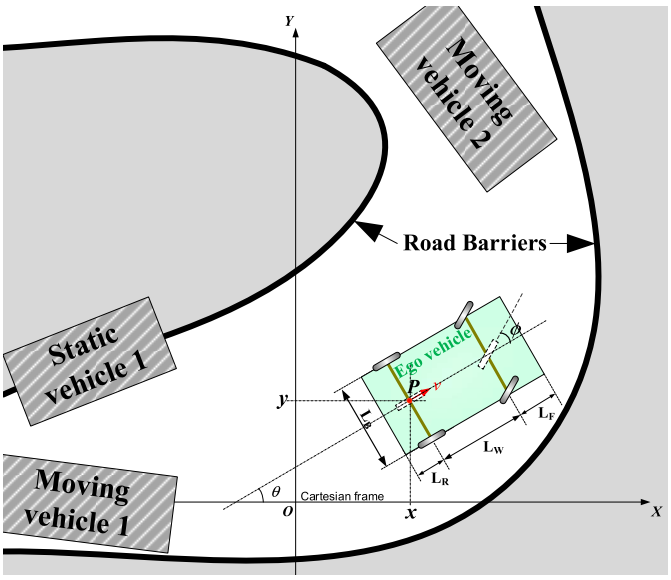


Fig. 3. Schematics on vehicle kinematics and a typical on-road driving scenario.

As the vehicle does not run fast on a curvy road, the bicycle model is sufficient to describe the vehicle mobility [30]:

$$\frac{\mathrm{d}}{\mathrm{d}t}\begin{bmatrix} x(t) \\ y(t) \\ \theta(t) \\ v(t) \\ \phi(t) \\ a(t) \end{bmatrix} = \begin{bmatrix} v(t)\cdot\cos\theta(t) \\ v(t)\cdot\sin\theta(t) \\ v(t)\cdot\tan\phi(t)/\mathrm{L_W} \\ a(t) \\ jerk(t) \end{bmatrix}, \quad t\in[0,\mathrm{T}], \tag{5}$$

where $\mathrm{L_W}$ denotes the wheelbase as marked in Fig. 3. Other parameters related to the vehicle geometrics, i.e., $\mathrm{L_F}$, $\mathrm{L_R}$, and $\mathrm{L_B}$, are depicted in Fig. 3 as well.

The inequalities $\mathbf{x} \le \mathbf{x}(t) \le \bar{\mathbf{x}}$ and $\mathbf{u} \le \mathbf{u}(t) \le \bar{\mathbf{u}}$ are presented as

$$\begin{bmatrix} \mathrm{a_{min}} \\ 0 \\ -\mathrm{jerk_{max}} \\ -\Omega_{\max} \\ -\Phi_{\max} \end{bmatrix} \le \begin{bmatrix} a(t) \\ v(t) \\ jerk(t) \\ \omega(t) \\ \phi(t) \end{bmatrix} \le \begin{bmatrix} \mathrm{a_{max}} \\ \mathrm{v_{max}} \\ \mathrm{jerk_{max}} \\ \Omega_{\max} \\ \Phi_{\max} \end{bmatrix}, \quad t\in[0,\mathrm{T}]. \tag{6}$$

2) Two-Point Boundary Constraints: $\mathbf{x}(0) = \mathbf{x}_{\mathrm{init}}, \mathbf{u}(0) = \mathbf{u}_{\mathrm{init}}, \mathbf{x}(T) = \mathbf{x}_{\mathrm{goal}}$, and $\mathbf{u}(T) = \mathbf{u}_{\mathrm{goal}}$ in (1) constitute the two-point boundary conditions. The state and control profiles at the initial moment $t = 0$ should be identified to reflect the ground truth at that moment:

$$\begin{aligned}&[x(0), y(0), \theta(0), v(0), a(0), jerk(0), \phi(0), \omega(0)] \\ &\quad = \left[\mathrm{x_0},\ \mathrm{y_0}, \theta_0,\ \mathrm{v_0},\ \mathrm{a_0},\ \mathrm{jerk_0}, \phi_0,\ \mathrm{w_0}\right].\end{aligned} \tag{7a}$$

By contrast, the boundary constraints at $t = \mathrm{T}$ are allowed to be set with more degrees of freedom because T is a future instance. In this paper, we only require that the ego vehicle runs in a stable status at $t = \mathrm{T}$, i.e.,

$$[a(\mathrm{T}), jerk(\mathrm{T}), \omega(\mathrm{T})] = [0, 0, 0]. \tag{7b}$$

3) Collision-Avoidance Constraints: Collision-avoidance constraints $h_{\mathrm{collision}} \le 0$ in (1) are used to prevent the ego vehicle from colliding with the road barriers and moving/static obstacles throughout [0,T]. Since the ego vehicle is rectangular, the collision-avoidance constraints are nominally non-differentiable [21], which are beyond the capability of a gradient-based OCP solver. To address this issue, a common solution is to construct a spatio-temporal corridor along the reference trajectory so that the ego vehicle is naturally separated from the surrounding obstacles and road barriers if it always stays in the corridor. Within-corridor constraints written in the following form are used to replace the nominal collision-avoidance ones:

$$\begin{aligned}&\mathrm{xlb}_i^j \le x^{\mathrm{disc}j}(t) \le \mathrm{xub}_i^j, \\ &\mathrm{ylb}_i^j \le y^{\mathrm{disc}j}(t) \le \mathrm{yub}_i^j, \\ &\qquad t\in[t_i, t_{i+1}], \quad i=0,\ldots,\mathrm{N_{FE}}-1, \ \ j=1,\ldots,\mathrm{N_{DISC}}.\end{aligned} \tag{8}$$

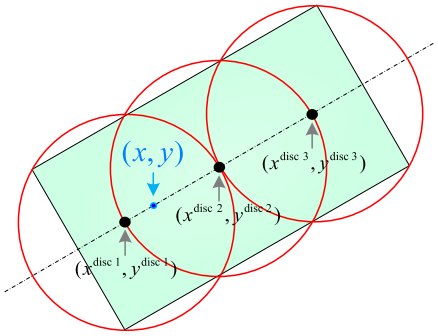


Fig. 4. Schematics on same-radius discs to cover the rectangular vehicle body ($\mathrm{N_{DISC}} = 3$ in this example).

As depicted in Fig. 4, the rectangular vehicle body is evenly covered by $\mathrm{N_{DISC}}$ same-radius discs, among which $(x^{\mathrm{disc}j}, y^{\mathrm{disc}j})$ denotes the location of the jth disc center. According to elementary geometrics, $(x^{\mathrm{disc}j}, y^{\mathrm{disc}j})$ is defined as

$$\begin{aligned}x^{\mathrm{disc}j}(t) &= x(t) + \left(\frac{2j-1}{2\mathrm{N_{DISC}}}\cdot(\mathrm{L_R}+\mathrm{L_W}+\mathrm{L_F}) - \mathrm{L_R}\right) \\ &\quad \cdot\cos\theta(t), \\ y^{\mathrm{disc}j}(t) &= y(t) + \left(\frac{2j-1}{2\mathrm{N_{DISC}}}\cdot(\mathrm{L_R}+\mathrm{L_W}+\mathrm{L_F}) - \mathrm{L_R}\right) \\ &\quad \cdot\sin\theta(t), \\ &\quad j=1,\ldots,\mathrm{N_{DISC}}, t\in[0,\mathrm{T}].\end{aligned} \tag{9}$$

Eq. (9) means that the location of each disc center is determined by $x(t)$, $y(t)$, and $\theta(t)$. The radius of each disc, i.e., $\mathrm{R^{disc}}$, is a constant related to $\mathrm{N_{DISC}}$:

$$\mathrm{R^{disc}} = \sqrt{\left(\frac{\mathrm{L_R}+\mathrm{L_W}+\mathrm{L_F}}{2\mathrm{N_{DISC}}}\right)^2 + \left(\frac{\mathrm{L_B}}{2}\right)^2}. \tag{10}$$

In (8), the temporal horizon [0,T] is divided into $\mathrm{N_{FE}}$ intervals $\{[t_i, t_{i+1}]\,|i = 0, 1, \ldots, \mathrm{N_{FE}} - 1\}$. For simplicity,

we require they are equidistant, that is,

$$t_0 = 0, \quad t_{\mathrm{N_{FE}}} = \mathrm{T}, \\ t_{i+1} - t_i = \mathrm{T}/\mathrm{N_{FE}}, \quad \forall i = 0, \ldots, \mathrm{N_{FE}} - 1. \tag{11}$$

Thus (8) requires that each of the $\mathrm{N_{DISC}}$ disc centers stays in an axis-aligned local box in each of the $\mathrm{N_{FE}}$ intervals. Since each local box is identified by 4 parameters, the entire within-corridor constraints require $4 \cdot \mathrm{N_{FE}} \cdot \mathrm{N_{DISC}}$ parameters, which are generated by the method introduced in Appendix B.

It deserves to note that there are alternative ways to build the corridor, but the one consisting of axis-aligned boxes has its unique advantage, which is elaborated in the next section.

As a summary of the whole section, the following OCP is formulated to describe the trajectory planning task for on-road autonomous driving:

$$\begin{aligned} &\text{Minimize(2)}, \\ &\text{s.t. Kinematic constraints (5) and (6);} \\ &\quad \text{Two-point boundary constraints (7);} \\ &\quad \text{Within-corridor constraints (8);} \\ &\quad \text{Definitions of disc centers (9).} \end{aligned} \tag{12}$$

Generally speaking, trajectory planning is about resolving the underlying conflicts between the kinematics-related and environment-related constraints. However, the within-corridor constraints (8) in OCP (12) are in question because they are formulated based on a coarsely searched reference trajectory, thus easily leaving out the free space necessary for kinematic feasibility. Therefore, directly solving OCP (12) does not always work. An alternative idea is to build an iterative framework, wherein the within-corridor constraints (8) are adaptively adjusted if they are found inappropriate. The details are introduced in the next section.

III. Trajectory Planning Method

Our proposed on-road trajectory planner is introduced in this section. The principle of the planner is presented first, followed by analyses of the proposal.

A. Principle of the Proposed Trajectory Planner

The overall principle of our proposed on-road trajectory planning method is presented as the following pseudo-codes in Alg. 1.

Alg. 1. An Iterative Trajectory Optimization Method

***Input*:** Reference trajectory $\mathbf{traj}^{\mathbf{coarse}}$;
***Output*:** Optimized trajectory $\mathbf{traj}^{\mathbf{optimized}}$;

1. $\boldsymbol{\chi} \leftarrow$ FormInitialGuess($\mathbf{traj}^{\mathbf{coarse}}$);
2. Initialize $\mathrm{w_{penalty}} \leftarrow w_{\mathrm{penalty0}}$, $iter \leftarrow 0$, $\mathbf{traj}^{\mathbf{optimized}} = \emptyset$;
3. **while** ($iter < \mathrm{iter_{max}}$), **do**
4. $\quad OCP \leftarrow$ FormulateIntemediateOCP($\boldsymbol{\chi}$);
5. $\quad \boldsymbol{\chi} \leftarrow$ SolveOCP(OCP, $\boldsymbol{\chi}$);
6. $\quad f_{\mathrm{penalty}}(T) \leftarrow$ MeasureInfeasibility($\boldsymbol{\chi}$);
7. $\quad$ **if** ($f_{\mathrm{penalty}}(T) < \varepsilon_{\mathrm{tol}}$), **then**
8. $\quad\quad \mathbf{traj}^{\mathbf{optimized}} \leftarrow$ ExtractTrajectoryFromSolutionVector($\boldsymbol{\chi}$);
9. $\quad\quad$ **return**;
10. $\quad$ **else**
11. $\quad\quad$ Update $\mathrm{w_{penalty}} \leftarrow w_{\mathrm{penalty}} \cdot \alpha$ and $iter \leftarrow iter + 1$;
12. $\quad$ **end if**
13. **end while**
14. **return**.

Alg. 1 begins with generating an initial guess via FormInitialGuess(), which loads the reference trajectory $\mathbf{traj}^{\mathbf{coarse}}$ derived by sampling and search, and then returns all the decision variables necessary for solving (12) numerically.

After the initialization of parameters and preparation for an initial guess in lines 1–2, Alg. 1 implements a while loop. In each iteration of the while loop, an intermediate OCP is formulated and solved numerically, the optimum of which serves as the initial guess for future usage.

In line 4, the function FormulateIntemediateOCP() is used to formulate an intermediate OCP, which is similar to (12) expect that the kinematics-related constraints are softened as external penalty costs before merged into the cost function (2). Concretely, the intermediate OCP is written as

$$\begin{aligned} &\text{Minimize } (2) + \mathrm{w_{penalty}} \cdot \mathrm{f_{penalty}}(T), \\ &\text{s.t. } \mathbf{x} \le \mathbf{x}(t) \le \bar{\mathbf{x}}, \quad \mathbf{u} \le \mathbf{u}(t) \le \bar{\mathbf{u}}; \\ &\quad \mathbf{x}(0) = \mathbf{x}_{\mathrm{init}}, \quad \mathbf{u}(0) = \mathbf{u}_{\mathrm{init}}, \quad \mathbf{x}(T) = \mathbf{x}_{\mathrm{goal}}, \\ &\quad \mathbf{u}(T) = \mathbf{u}_{\mathrm{goal}}; \quad \text{and (11).} \end{aligned} \tag{13}$$

Herein, $w_{penalty} > 0$ is a weighting parameter, and $f_{\mathrm{penalty}}(t)$ is defined as

$$f_{\mathrm{penalty}}(t) = \int_{\tau=0}^{t} \left(\|\dot{\mathbf{x}}(\tau) - f_{\mathrm{kinematics}}(\tau)\|^2 + \|g_{\mathrm{disc}}(\tau)\|^2 \right) \mathrm{d}\tau. \tag{14}$$

In (14), the vehicle kinematic constraints (5) are abstracted as $\dot{\mathbf{x}}(t) = f_{\mathrm{kinematics}}(t)$ while the disc center definitions (9) are abstracted as $g_{\mathrm{disc}}(t) = 0$.

SolveNLP(OCP, $\boldsymbol{\chi}$) solves the formulated intermediate OCP numerically with the warm-starting initial guess $\boldsymbol{\chi}$. The detailed procedures include discretizing the OCP into a nonlinear program (NLP) problem before solving it via a gradient-based local optimizer [31]. MeasureInfeasibility() outputs the kinematic infeasibility degree measured by $f_{\mathrm{penalty}}(T)$ according to (14). When $f_{\mathrm{penalty}}(T)$ is sufficiently close to 0^+, then ExtractTrajectoryFromSolutionVector($\boldsymbol{\chi}$) gets the trajectory from the solution vector. In this function, the decision variables used to represent a trajectory are extracted, which are regarded as skeletons to reformulate a temporarily continuous trajectory for the ego vehicle.

B. Property Analysis

This subsection leverages several properties of the proposed trajectory planner Alg. 1.

Firstly, the intermediate OCPs in Alg. 1 are different from one another because the within-corridor constraints are always updated according to the *latest* reference trajectories. The rationale behind this design is presented as follows. Recall that the corridors consist of axis-aligned local boxes, thus the free space would inevitably be left out in part, which renders a loss of optimality or even feasibility. Updating the corridors during the iterations can efficiently reduce the left-out free space if the

feasibility/optimality of the reference trajectory is improved during the iterations. At this point, using other types of convex polygons to form the corridors (e.g. in [23]) also suffers from a partial loss of free space because the free space is inherently irregularly shaped in our concerned task. By contrast, updating the corridors iteratively brings extra chances to reduce the loss of feasibility/optimality caused by the left-out free space.

Secondly, the feasibility/optimality of the reference trajectory is continuously improved during the iterations, which lays a foundation for the efficiency of the updating-corridor strategy mentioned in the preceding paragraph. Suppose that the corridors paved along the initial reference trajectory at $iter = 0$ are so poor that none kinematically feasible solutions lie in them. In such a case, recovering the kinematic feasibility naturally becomes a predominant scheme in minimizing the cost function of intermediate OCP (13). Therefore, the resultant optimum differs from the initial guess mainly in the reduction of the kinematic infeasibility. Although the kinematic infeasibility is not fully eliminated, the resultant trajectory is closer to being feasible in its shape, thus bringing about chances for further improvements in the subsequent iterations because the corridors can be updated then.

Thirdly, the intermediate OCPs are always feasible, which is an important foundation for the efficiency of the entire iterative framework. Suppose that the NLP problem derived by discretizing an intermediate OCP (13) is called NLP_{13}. It is interesting to note that NLP_{13} consists of a nonconvex cost function together with purely box constraints. Since all of the constraints are box constraints, NLP_{13} is always feasible unless the lower bounds of the box constraints exceed the upper bonds, which will not happen because the lower/upper bounds in our concerned task have real-world semantic senses.

Fourthly, the intermediate OCPs can be solved rapidly. Although the cost function of NLP_{13} is nonconvex, all the constraints are box constraints, which are the simplest type of linear constraints. Besides that, the initial guess of NLP_{13} is definitely a feasible starting point. This is because 1) it is an optimum derived in the preceding iteration, thus satisfying the box constraints there, and 2) the latest within-corridor constraints are constructed around the initial guess. Therefore, NLP_{13} is easy to solve. A prevalent idea in the community of autonomous driving trajectory planning is to simplify the OCP so that it can be discretized into a quadratic program (QP) problem rather than an NLP one that involves nonconvexity. However, formulating a QP usually requires linearizing the kinematic constraints. The linearized kinematic constraints easily become misleading if the Taylor expansion point is of low quality. In such cases, more iterations would be needed although the CPU runtime in every single iteration is small.

Fifthly, the corridor construction function runs fast. Axis-aligned boxes are chosen to form the corridors not only because they render box constraints, but also because they are much easier to identify than other types of convex polygons.

Sixthly, optimality is achieved if Alg. 1 exits from line 9. As the iteration continues, the kinematic infeasibility approaches 0^+, and the increase in $\text{w}_{\text{penalty}}$ (see line 11 of Alg. 1) could accelerate the process. When the kinematic infeasibility degree is relatively small, the polynomial (2) in the cost function of (13) becomes predominant, thus minimizing the cost function of (13) is close to minimizing the original cost function (2). The finally derived valid optimum minimizes (2) at the precision level of ε_{tol}.

Seventhly, the entire iterative optimization framework can be implemented in an anytime style if there is a strict limit on the CPU runtime. Since all the constraints in NLP_{13} are box constraints, the solution process of each NLP_{13} is always within the feasible region of the solution space. This property enables us to interrupt a gradient-based NLP solution process before its convergence without suffering from the risk of getting a worse trajectory, which stays out of the corridors or violates the kinematic constraints more severely. In this sense, even if the proposed planner does not find a valid optimum within predefined runtime limit or iterations, the result can still be passed on to the next planning frame for further improvements provided that the near-future segment of the resultant trajectory is found to be kinematically feasible.

IV. Experimental Results and Discussions

Simulations and experiments are conducted to show the efficiency, solution speed, and closed-loop tracking performance of the proposed trajectory planner. Concretely, simulations are deployed to show our proposal can outperform other state-of-the-art planners in the same community whereas experiments are conducted to show the planned trajectories are easy to track and the planning runtime is short.

Typical simulation and experimental results are reported in https://www.bilibili.com/video/BV1X54y1m7mr/.

A. Simulation Setup

Simulations are executed on an i9-9900 CPU that runs at 3.10×2GHz. An open-source NLP solver called IPOPT [32] is adopted in the MATLAB+AMPL environment [33]. Basic parametric settings are listed in Table I.

A typical curvy road scenario containing a right-turn and two U-turns is defined, wherein the social vehicles are randomly set as static or moving. The velocity of each moving vehicle is assumed to be constant, which is determined randomly.

B. On the Efficiency of the Proposed Planner

This subsection shows the performance of our proposed trajectory planner when tested on a simple case with 6 randomly defined static obstacles. The optimized trajectory, together with the footprints are depicted in Fig. 5, which indicates that the trajectory is collision-free. Fig. 6 shows the velocity and steering angle profiles associated with the optimized trajectory, which stay in the allowable bounds and appear to be smooth.

To see the details in the iterative framework of Alg. 1, Fig. 7 depicts the optimized trajectory together with the intermediate OCP solutions. In this figure, the reference trajectory derived by sampling and search can be taken as the intermediate trajectory at iteration 0. The reference trajectory is refined in 2 extra iterations. Since the reference trajectory in each iteration is different, our efforts to re-construct the corridors make sense.

TABLE I
PARAMETRIC SETTINGS FOR SIMULATIONS

Parameter	Description and unit	Value
L_F	Front hang length of the ego vehicle	0.96
L_W	Wheelbase of the ego vehicle (m)	2.80
L_R	Rear hang length of the ego vehicle	0.929
L_B	Width of the ego vehicle (m)	1.942
v_{max}	Upper bound of $v(t)$ (m/s)	12.0
a_{min}, a_{max}	Lower and upper bounds of $a(t)$	-5.0, 5.0
$jerk_{max}$	Upper bound of $\lvert jerk(t)\rvert$ (m/s^3)	10.0
Φ_{max}	Upper bound of $\lvert\phi(t)\rvert$ (rad)	0.85
Ω_{max}	Upper bound of $\lvert\omega(t)\rvert$ (rad/s)	1.5
w_u	Weighting parameter in Eq. (2)	0.5
$w_{r\theta}$	Weighting parameter in Eq. (3)	2.0
w_{rw}	Weighting parameter in Eq. (4)	1.0
T	Time horizon length (s)	16.0
N_{DISC}	Number of discs used to cover the vehicle body	2
N_{FE}	Number of finite elements used to discretize an OCP	320
$iter_{max}$	Maximum iteration number in Alg. 1	5
$w_{penalty\,0}$	Initial value of weighting parameter $w_{penalty}$	10^5
α	Multiplier to enlarge $w_{penalty}$ during the iterations	10
ε_{tol}	Violation tolerance w.r.t. the softened nonlinear constraints	10^{-4}

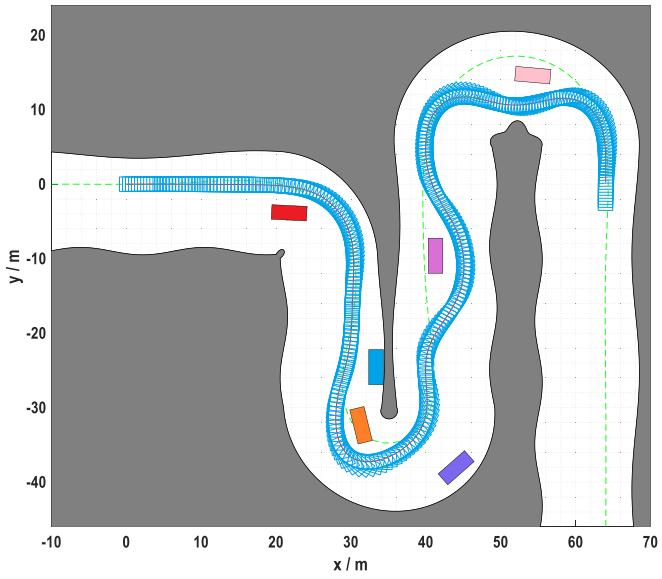


Fig. 5. Optimized trajectory and footprints derived by using Alg. 1 to handle a simulation case with 6 static obstacles. This figure can be seen more clearly if zoomed in.

To leverage the effect of the iterative framework, a comparison is made with simply solving OCP (12) for once. However, the numerical solution process fails after consuming 5.374s. By contrast, Alg. 1 only takes 0.378s to output an optimized trajectory. This comparison clearly shows that deploying an iterative framework is efficient to get rid of the low-quality

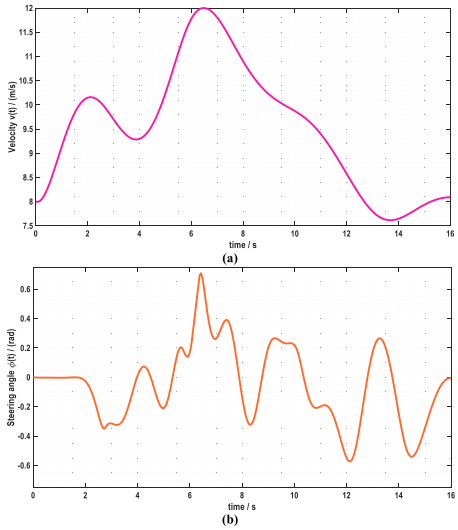


Fig. 6. The velocity and steering angle profiles in association with the optimized trajectory illustrated in Fig. 5.

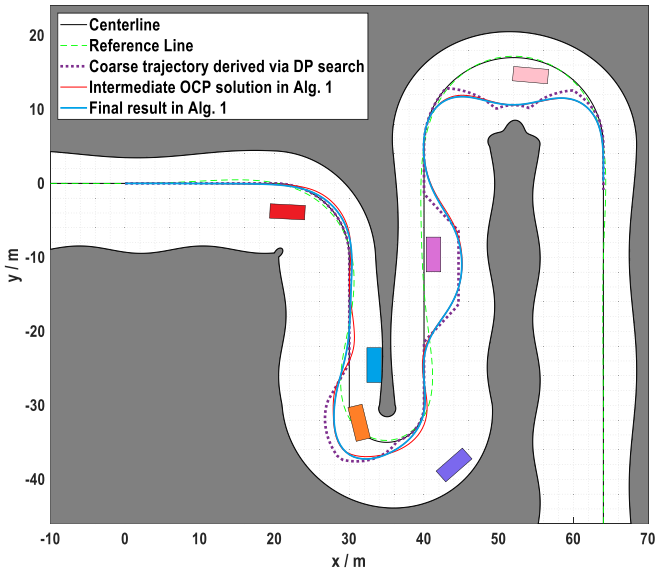


Fig. 7. Intermediate OCP solutions when Alg. 1 is adopted to handle the simulation case mentioned in Fig. 5.

reference trajectory and the side effects of the corridors initially.

To evaluate the feasibility of the final output of Alg. 1, we define a variant of Alg. 1 (denoted as Alg. 2), which is the same as Alg. 1 except that the softened nonlinear constraints in the last intermediate OCP are formulated in their nominal forms again, i.e., hard constraints. Through this, the output of Alg. 2 would strictly satisfy all of the constraints. The results of Algs. 1 and 2 are illustrated in Fig. 8, which shows that the results differ at a level of 0.01 m. Recall that the output of our proposed planner violates the softened nonlinear constraints at most by ε_{tol}, thus the output of Alg. 1 is close to being really optimal if ε_{tol} is set sufficiently small. In using Alg. 1, the infeasibility criterion $f_{\text{penalty}}(T)$ changes from 7.02×10^7, 3.85×10^{-2} to 2.73×10^{-5}, which is at an acceptable

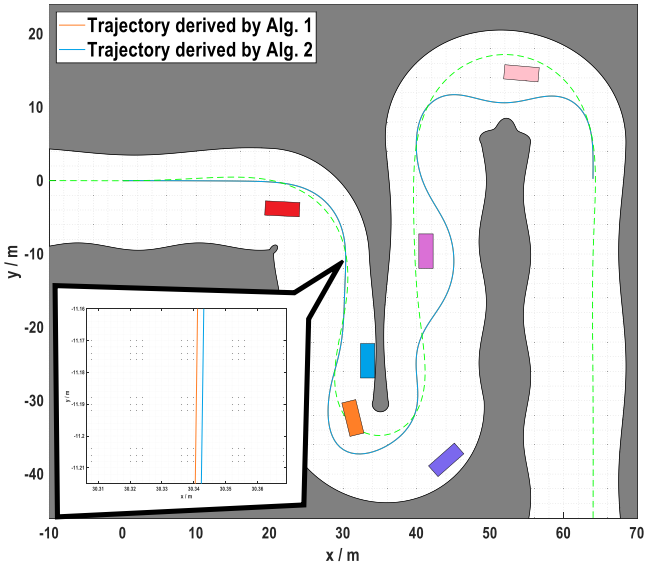


Fig. 8. Comparative simulation results to show the feasibility of the proposed trajectory planner's outputs.

level finally. Besides that, it deserves to note that Alg. 2 takes 0.171s longer than Alg. 1, which means that requiring each intermediate OCP purely contains box constraints is an effective way to reduce the runtime.

As a conclusion of this subsection, designing an iterative framework in Alg. 1 has validated intention and has obvious benefits in solution efficiency and real-time performance.

C. Comparison With a Frenet-Based Planner

This subsection investigates the benefit of planning in the Cartesian frame. For comparison, a well-known on-road trajectory planner in the Frenet frame called Apollo EM planner [14] is adopted as the competitor. The EM planner is featured by implementing path planning and velocity planning alternately until a converged trajectory is derived. In either EM planner or our proposed Alg. 1, the cost function is about minimizing the control profiles and encouraging the optimized trajectory close to the coarse trajectory derived by DP. The resultant trajectories have minor differences as shown in Fig. 9a. Particularly, the EM planner consumes only 0.046s to finish, which is quite fast. To take a further insight into the results, we evaluate the trajectory curvature via the following criterion:

$$\kappa(s) = \frac{x'(s)y''(s) - y'(s)x''(s)}{\left[(x'(s))^2 + (y'(s))^2\right]^{\frac{3}{2}}}, \tag{15}$$

where s denotes the mileage profile along a trajectory. Eq. (15) yields a relationship between κ and ϕ: $\kappa = (\tan\phi)/\mathrm{L_W}$, thus $\kappa(s)$ is boundary constraints:

$$-\frac{\tan(\Phi_{\max})}{\mathrm{L_W}} \le \kappa(s) \le \frac{\tan(\Phi_{\max})}{\mathrm{L_W}}. \tag{16}$$

The curvature profiles of the trajectories derived by both planners, together with the bounds defined in (16), are plotted in Fig. 9b. The EM planner's result exceeds the curvature limit, which typically shows the fact that a Frenet-based planner is

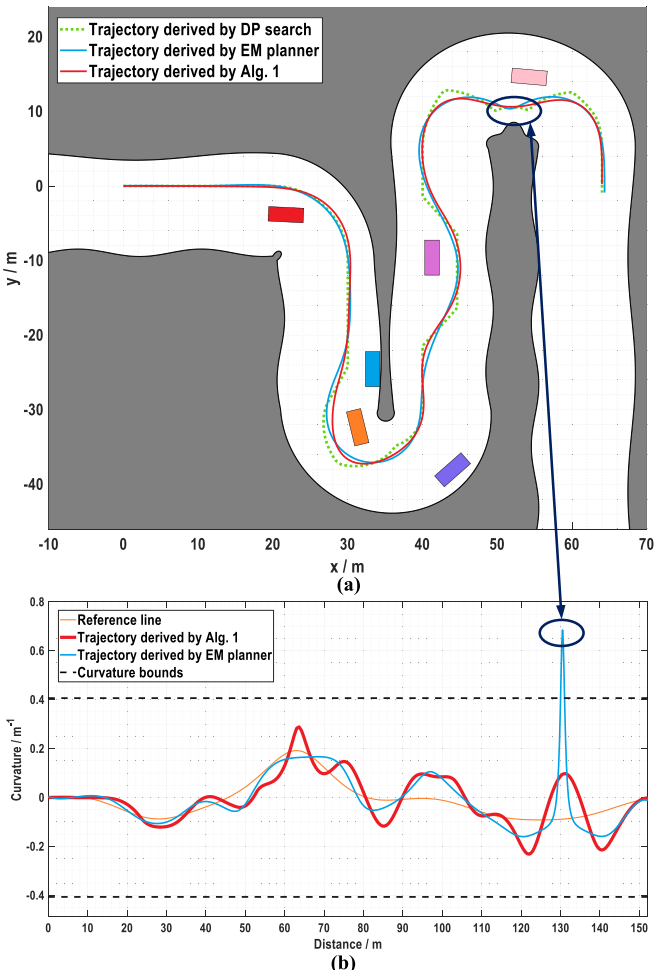


Fig. 9. Comparative simulation results to show the superiority of Alg. 1 against a Frenet-based trajectory planner: (a) trajectories planned by Alg. 1 and EM planner; and (b) curvature profiles in association with the planned trajectories.

not aware of the kinematic principle of the ego vehicle, which actually lives in the real-world Cartesian frame. By contrast, our proposal does not suffer from this sort of issue.

D. Comparison With Other Cartesian-Based Planners

In this subsection, Alg. 1 is compared with an existing Cartesian-based planner proposed by Liniger *et al.* [27], which is denoted as an MPC-based planner. Like Alg. 1, the MPC-based planner also facilitates the trajectory planning process by decoupling the vehicle kinematic constraints and collision-avoidance constraints, which are nominally coupled together. In more details, the MPC-based method linearizes the nonlinear kinematic constraints, relaxes the collision-avoidance constraints via the introduction of extra slack variables, and thus builds an OCP with the slack variables minimized. The OCP is iteratively solved until convergence.

Consequently, the MPC-based method takes 5 iterations to complete. The runtime of the MPC-based method is 0.956s, which is much longer than our Alg. 1. The reason for the low speed of the MPC-based method may be that the optimization in each iteration is only done in a small-scale trust region, thus the step length in the gradient-based optimization

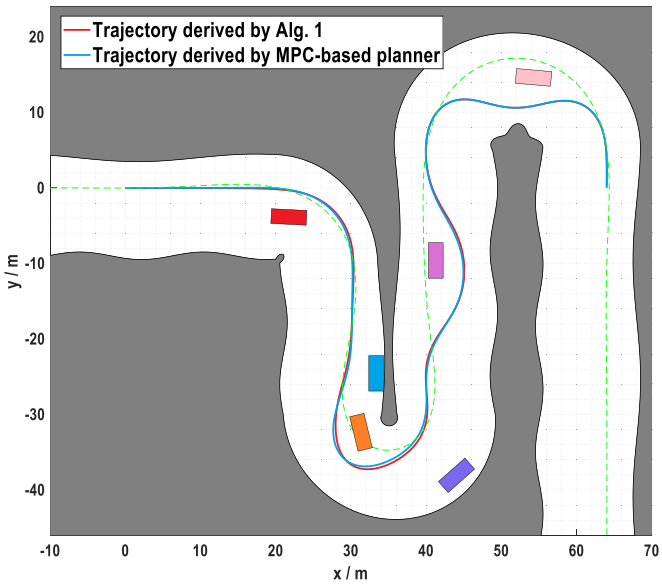


Fig. 10. Comparative simulation results to show the superiority of Alg. 1 against another Cartesian-based trajectory planner.

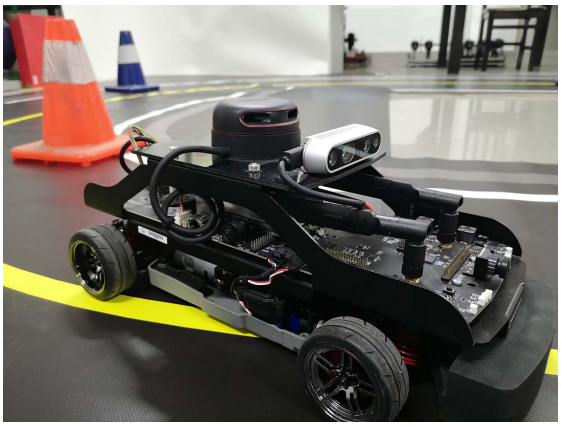


Fig. 11. QCar: a small-size autonomous vehicle platform.

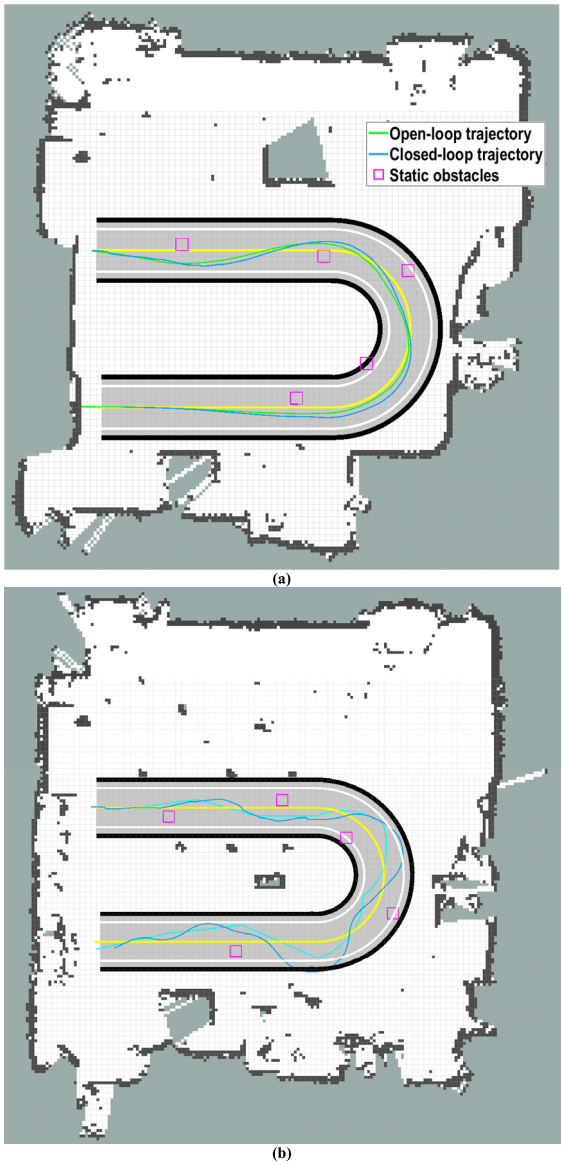


Fig. 12. Closed-loop tracking performances tested on the QCar platform in an in-door U-shape road scenario: (a) tracking the optimized trajectories provided by Alg. 1; and (b) tracking the reference trajectories provided by DP search.

becomes small, which calls for more iterations. Like most of the MPC-like methods that linearize the vehicle kinematics, getting a good solution relies highly on a near-feasible Taylor expansion point. More iterations would be needed if a good Taylor expansion point is not available initially.

Another side effect of using the MPC-based planner is that the slack variables enable to alter the homotopy class during the iterations, which brings about risks to make the planned trajectory inconsistent with the decision made in a higher-level module. Similar to the MPC approach, there have been large numbers of sequential QP (SQP) or sequential convex program (SCP) based methods with linearization operations. The proposed method shows that making each intermediate OCP contains purely box constraints has its merits when compared with the predominant solution facilitation strategies.

E. Experimental Setup and Results

In addition to the aforementioned simulations, real-world experiments were also conducted to investigate the tractability of the trajectories planned by Alg. 1 as well as its time efficiency.

Experiments were conducted on a small-size autonomous vehicle platform QCar produced by Quanser® (Fig. 11). The signals from a single-beam LiDAR and an IMU are integrated for indoor localization [34]. The LiDAR is also responsible for static obstacle detection (no moving obstacles are deployed for the indoor experiments, thus the obstacle prediction module can be safely discarded). The proposed trajectory planning method is written in C++ for fast onboard implementation. Regarding the closed-loop control module, a P-controller tracks in the longitudinal direction whereas an LQR-controller tracks in the lateral direction. A U-shape road scenario is set up, wherein multiple cone barrels are placed for the small vehicle to evade.

A typical result is depicted in Fig. 12, wherein the open-loop trajectory is planned in a receding horizon way (the duration

of each planning horizon is set to 1000ms). As shown in Fig. 12a, the open-loop and closed-loop trajectories differ not much, which indicates that the trajectories proposed by our proposed Alg. 1 are easy to track. As a comparison, we sent the reference trajectory roughly searched by DP to the low-level controllers. The result depicted in Fig. 12b indicates that a kinematically infeasible trajectory would be difficult to track. To conclude, the trajectory planning resulted derived by Alg. 1 are easy to track.

Among the planning frames from the starting instance to the end, the DP search procedure consumed 0.732ms, and Alg. 1 consumed 273.630ms on average. Either runtime is well below the entire planning horizon length, which indicates that the proposed planner is capable of handling on-road trajectory planning tasks in the real world.

V. Conclusion

This paper has proposed an optimization-based trajectory planning method in the Cartesian frame for autonomous driving on a curvy road. In contrast with the prevalent Frenet-based planners that run fast but cannot accurately reflect the vehicle kinematics, our proposed planner models the vehicle kinematics and collision avoidance in the real-world Cartesian frame where the ego vehicle actually lives in. The choice of Cartesian frame renders several challenges in the problem formulation and problem-solving procedures, which are systematically addressed by the proposed iterative optimization framework. Simulations have been conducted to show that 1) the Frenet frame has its limitations to describe the vehicle kinematics, 2) a Cartesian-based planner is promising to provide accurate solutions rapidly if being well designed, and 3) the predominant optimization facilitation strategies that commonly involve convexification or linearization via Taylor expansion points are not perfect in the runtime.

It deserves to emphasize here that a Frenet-based planner does have its natural advantages to describe the traffic-rule-related constraints on the road [24] while modeling these constraints is not easy for a Cartesian-based planner. Our future work is focused on how to describe the spatially constrained on-road autonomous driving scenarios within a Cartesian-based trajectory planner. Also, it deserves to develop a unified method that integrates both the Cartesian and Frenet frames and switches between them adaptively as the on-road scenario changes.

Appendix A

This section lists the typical disadvantages in modeling the trajectory planning problems in the Frenet frame.

A. Excessive Reliance on Reference Line

The establishment of a Frenet frame relies on a reference line to reflect the road trend. The reference line should be curvature- continuous; otherwise, even a curvature-continuous trajectory derived in the Frenet frame becomes curvature-discontinuous after being converted back into the Cartesian frame. This argument is validated by the following analysis. Suppose that the curvature of a trajectory in the Frenet frame is κ_{f}, and the curvature becomes κ_{x} after the trajectory gets converted back to the Cartesian frame. According to [7], the relationship between κ_{f} and κ_{x} is written as

$$\kappa_{\mathrm{x}} = \left\{ \frac{\left[\kappa_{\mathrm{f}} + (\kappa_{\mathrm{r}}' l + \kappa_{\mathrm{r}} l') \tan \Delta\theta\right] \cos^2 \Delta\theta}{1 - \kappa_{\mathrm{r}} l} + \kappa_{\mathrm{r}} \right\} \cdot \frac{\cos \Delta\theta}{1 - \kappa_{\mathrm{r}} l}, \tag{A1}$$

where κ_{r} denotes the curvature of the reference line, and l denotes the lateral offset of a specified point P (see Fig. 13a). The coordinate value of P in the Frenet frame is (s, l). Q denotes the mapping point along the reference line, whose coordinate value is $(s, 0)$. Moreover, we have $l' \equiv \mathrm{d}l/\mathrm{d}s$, and $\Delta\theta \equiv \theta_{\mathrm{x}} - \theta_{\mathrm{r}}$, where θ_{x} denotes the orientation angle of the tangential line at P, and θ_{r} denotes the orientation angle of the tangential line at Q. Notably, κ_{x} is determined by both κ_{f} and κ_{r}. Thus, if κ_{r} is discontinuous, then κ_{x} is discontinuous regardless of whether κ_{f} is continuous or not. This analysis clearly shows that the establishment of a Frenet frame relies on a curvature-continuous reference line. If such a reference line fails to be found, Frenet-based planners become inapplicable.

B. Inconsistent Conversion From Cartesian to Frenet Frame

The coordinate conversion from the Cartesian frame to the Frenet frame is valid when the lateral offset away from the reference line is smaller than a threshold $1/\kappa_{\mathrm{r}}$ [35]–[37]. However, the conversion is not always unique, the reason is given as follows. Suppose the reference line is presented by a set of waypoints $\mathbf{ref} = \left\{(x_i^{\mathrm{w}}, y_i^{\mathrm{w}})\right\}$. For a point $P = (x_{\mathrm{p}}, \mathrm{y}_{\mathrm{p}})$ in the 2D space, a matching point P_{ref} is defined as the one being closest to P along the reference line, i.e.,

$$P_{\mathrm{ref}} = \underset{(x_i^{\mathrm{w}}, \mathrm{y}_i^{\mathrm{w}}) \in \mathbf{ref}}{\arg\min} \left\| (x_i^{\mathrm{w}} - x_{\mathrm{p}}, \mathrm{y}_i^{\mathrm{w}} - \mathrm{y}_{\mathrm{p}}) \right\|. \tag{A2}$$

Nevertheless, multiple matching points satisfying this criterion may exist. As shown in Fig. 13b, all the mapping points marked in red are equally closest to P_0, which causes difficulty to identify s dimension value of P_0 when converted into the Frenet frame. The Frenet frame is imperfect even when the conversion is unique. In Fig. 13b, P_1 and P_2 differ drastically in their s dimension values, although they are close with each other in the Cartesian frame. This means that a spatially continuous trajectory in the Cartesian frame would become discontinuous when converted into the Frenet frame. To summarize, the mapping from the Cartesian frame to the Frenet frame is neither unique nor uniform, thereby making the conversion not stable.

1) Ignorance of Vehicle Shape Distortion: When a road in the Cartesian frame is standardized as a tube in the Frenet frame, the ego vehicle's shape would no longer be a rectangle after the frame conversion. Regarding the vehicle shape, the gap between the two frames would be large if the road is curvy. However, quite few studies have considered this problem [38].

2) Disability to Support Cooperative Planning: Since a Frenet frame is built along a single reference line, it cannot support cooperative planning for multiple vehicles if they have different reference lines. Thus prevailing intelligent transportation functions such as cooperative lane change and

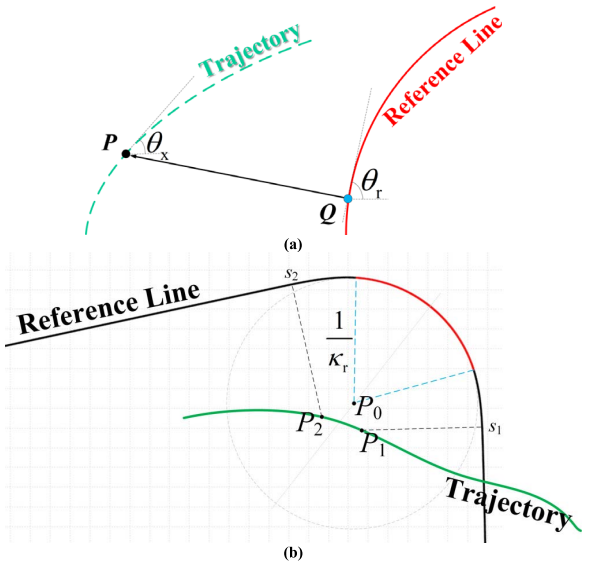


Fig. 13. Schematics on the conversion between Cartesian and Frenet frames: (a) geometric notations related to reference line and a to-be-converted trajectory; and (b) a typical case reflects the trajectory discontinuity caused by frame conversion.

autonomous intersection management cannot be realized in the Frenet frame [39], [40].

APPENDIX B

This section presents the principle to identify the local boxes which are used to construct a corridor [41].

Suppose that the reference trajectory is denoted as $\left(\mathbf{traj}^{\mathbf{coarse}}.x(t), \mathbf{traj}^{\mathbf{coarse}}.y(t), \mathbf{traj}^{\mathbf{coarse}}.\theta(t)\right), t \in [0, \mathrm{T}]$. As many as $\mathrm{N_{FE}}$ waypoints $\mathbf{W} = \{[x_i, y_i, \theta_i] \,|i = 1, \ldots, \mathrm{N_{FE}}\}$ are identified by sampling the midpoint of each time interval $[t_i, t_{i+1}]$ in $\mathbf{traj}^{\mathbf{coarse}}$, where $\{t_i | \, i = 0, \ldots, \mathrm{N_{FE}}\}$ is defined in (10).

With each waypoint $[x_i, y_i, \theta_i]$ at hand, one may identify the locations of the $\mathrm{N_{DISC}}$ disc centers via (9). Let us denote the location of the jth disc center in association with the ith waypoint as $(x_i^{\mathrm{disc}\,j}, y_i^{\mathrm{disc}\,j})$. The total number of the disc centers is $\mathrm{N_{FE}} \cdot \mathrm{N_{DISC}}$. We need to construct a local box around each of the $\mathrm{N_{FE}} \cdot \mathrm{N_{DISC}}$ disc centers. Without loss of generality, let us focus on how to identify the local box around the jth disc center associated with the ith waypoint.

Recall that the ith waypoint is in association with the time interval $t \in [t_i, t_{i+1}]$. The footprints of the perceived static obstacles and predicted moving obstacles during $[t_i, t_{i+1}]$, together with the road barriers, are plotted in a single map (Fig. 14a). In that map, the occupancy grids refer to the locations that the ego vehicle has to evade during $t \in [t_i, t_{i+1}]$. A new map (later it is called a *dilated map*) is formed by inflating the occupancy grids in that map by $\mathrm{R^{disc}}$ (Fig. 14b). Nominally, $(x_i^{\mathrm{disc}\,j}, y_i^{\mathrm{disc}\,j})$ should lie in the blank space of the *dilated map*. If this is not the case, an extra procedure is executed to find an alternative core point close to $(x_i^{\mathrm{disc}\,j}, y_i^{\mathrm{disc}\,j})$. As shown in Fig. 14c, we build an involute centered at $(x_i^{\mathrm{disc}\,j}, y_i^{\mathrm{disc}\,j})$, checks whether each trial point is collision-free sequentially because a good one is found. When

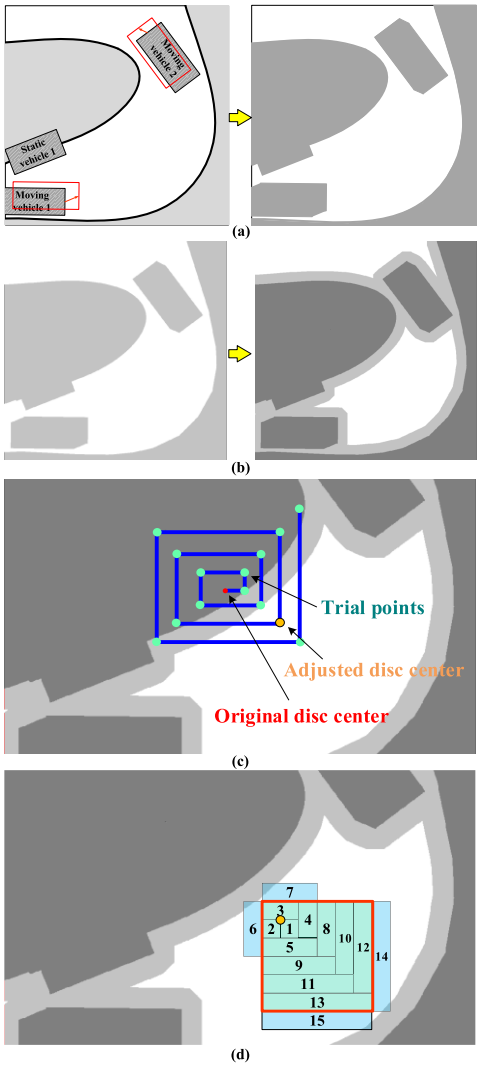


Fig. 14. Schematics on the principle to construct corridors: (a) a static map containing footprints of obstacles in a short period; (b) map dilation; (c) searching process for an alternative disc center; (d) expansion process to identify a local box.

a valid core point is available, the local box can be identified by repeatedly checking whether the incremental expansions in the four axis-aligned directions are valid (Fig. 14d); a valid expansion is merged into the main local box region whereas an invalid one is discarded and no further trial in the corresponding direction is conducted. The finally derived local box is axis-aligned, which is represented by the boundary values in the x and y axes: $(\mathrm{xlb}_i^j, \mathrm{xub}_i^j) \times (\mathrm{ylb}_i^j, \mathrm{yub}_i^j)$.

The remaining local boxes can be identified in the same way. All the local boxes form a spatio-temporal corridor along the reference trajectory $\mathbf{traj}^{\mathbf{coarse}}$.

REFERENCES

[1] L. Claussmann, M. Revilloud, D. Gruyer, and S. Glaser, "A review of motion planning for highway autonomous driving," *IEEE Trans. Intell. Transp. Syst.*, vol. 21, no. 5, pp. 1826–1848, May 2020.

[2] J. Guo, U. Kurup, and M. Shah, "Is it safe to drive? An overview of factors, metrics, and datasets for driveability assessment in autonomous driving," *IEEE Trans. Intell. Transp. Syst.*, vol. 21, no. 8, pp. 3135–3151, Aug. 2020.

[3] S. Luo, X. Li, and Z. Sun, "An optimization-based motion planning method for autonomous driving vehicle," in *Proc. 3rd Int. Conf. Unmanned Syst. (ICUS)*, Nov. 2020, pp. 739–744.

[4] B. Paden, M. Čáp, S. Z. Yong, D. Yershov, and E. Frazzoli, "A survey of motion planning and control techniques for self-driving urban vehicles," *IEEE Trans. Intell. Veh.*, vol. 1, no. 1, pp. 33–55, Mar. 2016.

[5] B. Li, K. Wang, and Z. Shao, "Time-optimal maneuver planning in automatic parallel parking using a simultaneous dynamic optimization approach," *IEEE Trans. Intell. Transp. Syst.*, vol. 17, no. 11, pp. 3263–3274, Nov. 2016.

[6] C. Katrakazas, M. Quddus, W.-H. Chen, and L. Deka, "Real-time motion planning methods for autonomous on-road driving: State-of-the-art and future research directions," *Transp. Res. C, Emerg. Technol.*, vol. 60, pp. 416–442, Nov. 2015.

[7] M. Werling, J. Ziegler, S. Kammel, and S. Thrun, "Optimal trajectory generation for dynamic street scenarios in a Frenét frame," in *Proc. IEEE Int. Conf. Robot. Automat. (ICRA)*, May 2010, pp. 987–993.

[8] X. Li, Z. Sun, D. Cao, Z. He, and Q. Zhu, "Real-time trajectory planning for autonomous urban driving: Framework, algorithms, and verifications," *IEEE/ASME Trans. Mechatronics*, vol. 21, no. 2, pp. 740–753, Apr. 2016.

[9] H. Fan *et al.*, "Baidu Apollo EM motion planner," 2018, *arXiv:1807.08048*.

[10] Y. Wang, S. Li, W. Cheng, X. Cui, and B. Su, "Toward efficient trajectory planning based on deterministic sampling and optimization," in *Proc. Chin. Automat. Congr. (CAC)*, Nov. 2020, pp. 1318–1323.

[11] B. Li *et al.*, "On-road trajectory planning with spatio-temporal RRT* and always-feasible quadratic program," in *Proc. IEEE 16th Int. Conf. Autom. Sci. Eng. (CASE)*, Aug. 2020, pp. 943–948.

[12] W. Lim, S. Lee, M. Sunwoo, and K. Jo, "Hierarchical trajectory planning of an autonomous car based on the integration of a sampling and an optimization method," *IEEE Trans. Intell. Transp. Syst.*, vol. 19, no. 2, pp. 613–626, Feb. 2018.

[13] W. Lim, S. Lee, M. Sunwoo, and K. Jo, "Hybrid trajectory planning for autonomous driving in on-road dynamic scenarios," *IEEE Trans. Intell. Transp. Syst.*, vol. 22, no. 1, pp. 341–355, Jan. 2021.

[14] S. Manzinger, C. Pek, and M. Althoff, "Using reachable sets for trajectory planning of automated vehicles," *IEEE Trans. Intell. Vehicles*, vol. 6, no. 2, pp. 232–248, Jun. 2021.

[15] 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, Sep. 2009.

[16] L. Ma, J. Xue, K. Kawabata, J. Zhu, C. Ma, and N. Zheng, "Efficient sampling-based motion planning for on-road autonomous driving," *IEEE Trans. Intell. Transp. Syst.*, vol. 16, no. 4, pp. 1961–1976, Aug. 2015.

[17] X. Li, Z. Sun, D. Cao, D. Liu, and H. He, "Development of a new integrated local trajectory planning and tracking control framework for autonomous ground vehicles," *Mech. Syst. Signal Process.*, vol. 87, pp. 118–137, Mar. 2017.

[18] C. Sun, Q. Li, and L. Li, "A gridmap-path reshaping algorithm for path planning," *IEEE Access*, vol. 7, pp. 183150–183161, 2019.

[19] J. Chen, W. Zhan, and M. Tomizuka, "Autonomous driving motion planning with constrained iterative LQR," *IEEE Trans. Intell. Veh.*, vol. 4, no. 2, pp. 244–254, Jun. 2019.

[20] J. Chen, C. Liu, and M. Tomizuka, "FOAD: Fast optimization-based autonomous driving motion planner," in *Proc. Annu. Amer. Control Conf. (ACC)*, Jun. 2018, pp. 4725–4732.

[21] 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, Sep. 2015.

[22] J. Ziegler, P. Bender, T. Dang, and C. Stiller, "Trajectory planning for bertha—A local, continuous method," in *Proc. IEEE Intell. Vehicles Symp. (IV)*, Jun. 2014, pp. 450–457.

[23] C. Liu, C. Lin, and M. Tomizuka, "The convex feasible set algorithm for real time optimization in motion planning," *SIAM J. Control Optim.*, vol. 56, no. 4, pp. 2712–2733, Jun. 2018.

[24] W. Ding, L. Zhang, J. Chen, and S. Shen, "Safe trajectory generation for complex urban environments using spatio-temporal semantic corridor," *IEEE Robot. Autom. Lett.*, vol. 4, no. 3, pp. 2997–3004, Jul. 2019.

[25] B. Li and Y. M. Zhang, "Fast trajectory planning for off-road autonomous driving with a spatiotemporal tunnel and numerical optimal control approach," in *Proc. IEEE 4th Int. Conf. Adv. Robot. Mechatronics (ICARM)*, Jul. 2019, pp. 924–929.

[26] Y. Zhang *et al.*, "Hybrid trajectory planning for autonomous driving in highly constrained environments," *IEEE Access*, vol. 6, pp. 32800–32819, 2018.

[27] A. Liniger, A. Domahidi, and M. Morari, "Optimization-based autonomous racing of 1:43 scale RC cars," *Optim. Control Appl. Methods*, vol. 36, no. 5, pp. 628–647, 2015.

[28] M. McNaughton, C. Urmson, J. M. Dolan, and J.-W. Lee, "Motion planning for autonomous driving with a conformal spatiotemporal lattice," in *Proc. IEEE Int. Conf. Robot. Automat. (ICRA)*, May 2011, pp. 4889–4895.

[29] W. Xu, J. Pan, J. Wei, and J. M. Dolan, "Motion planning under uncertainty for on-road autonomous driving," in *Proc. IEEE Int. Conf. Robot. Automat. (ICRA)*, May 2014, pp. 2507–2512.

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

[31] B. Li and Z. Shao, "Simultaneous dynamic optimization: A trajectory planning method for nonholonomic car-like robots," *Adv. Eng. Softw.*, vol. 87, pp. 30–42, Sep. 2015.

[32] A. Wächter and L. T. 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, May 2006.

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

[34] M. Jaimez, J. G. Monroy, and J. Gonzalez-Jimenez, "Planar odometry from a radial laser scanner. A range flow-based approach," in *Proc. IEEE Int. Conf. Robot. Automat. (ICRA)*, May 2016, pp. 4479–4485.

[35] F. Bayer and J. Hauser, "Trajectory optimization for vehicles in a constrained environment," in *Proc. IEEE 51st IEEE Conf. Decis. Control (CDC)*, Dec. 2012, pp. 5625–5630.

[36] H. Wang, J. Kearney, and K. Atkinson, "Robust and efficient computation of the closest point on a spline curve," in *Proc. 5th Int. Conf. Curves Surf. (ICCS)*, 2002, pp. 397–406.

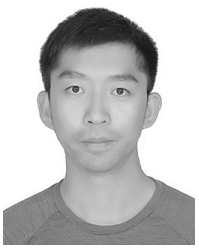
[37] T. D. Barfoot and C. M. Clark, "Motion planning for formations of mobile robots," *Robot. Auton. Syst.*, vol. 46, no. 2, pp. 65–78, Feb. 2004.

[38] R. Oliveira, P. F. Lima, G. C. Pereira, J. Martensson, and B. Wahlberg, "Path planning for autonomous bus driving in highly constrained environments," in *Proc. IEEE Intell. Transp. Syst. Conf. (ITSC)*, Oct. 2019, pp. 2743–2749.

[39] B. Li, Y. Zhang, Y. Ge, Z. Shao, and P. Li, "Optimal control-based online motion planning for cooperative lane changes of connected and automated vehicles," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Sep. 2017, pp. 3689–3694.

[40] B. Li, Y. Zhang, T. Acarman, Y. Ouyang, C. Yaman, and Y. Wang, "Lane-free autonomous intersection management: A batch-processing framework integrating reservation-based and planning-based methods," in *Proc. IEEE Int. Conf. Robot. Automat. (ICRA)*, May 2021, pp. 7915–7921.

[41] 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. 8, 2021, doi: 10.1109/TITS.2021.3109011.



Bai Li (Member, IEEE) received the B.S. degree from the School of Advanced Engineering, Beihang University, China, in 2013, and the Ph.D. degree from the College of Control Science and Engineering, Zhejiang University, China, in 2018. From November 2016 to June 2017, he was a joint training Ph.D. student with the Department of Civil and Environmental Engineering, University of Michigan, Ann Arbor, USA. From 2018 to 2020, he was an Algorithm Engineer at the JDX Research and Development Center of Automated Driving, JD Inc., China. He is currently an Associate Professor with the College of Mechanical and Vehicle Engineering, Hunan University, China. He is the first author of nearly 60 journals/conference papers and two books in the community of robotics. His research interest includes motion planning of automated vehicles. He was a recipient of the 2014–2016 Best Journal Paper Prize (first author of the awarded paper) from the International Federation of Automatic Control (IFAC). Since 2022, he has been an Associate Editor of the IEEE TRANSACTIONS ON INTELLIGENT VEHICLES.

Yakun Ouyang (Graduate Student Member, IEEE) received the B.S. degree from the School of Information Engineering, Nanchang University, Nanchang, China, in 2020. He is currently pursuing the master's degree with the College of Mechanical and Vehicle Engineering, Hunan University, China. His research interests include decision making, trajectory planning, control, and software engineering of an autonomous vehicle system. He was a First-Prize Recipient of the 2019 National University Students Intelligent Car Race.

Li Li (Fellow, IEEE) is currently an Associate Professor with the Department of Automation, Tsinghua University, Beijing, China, working in the fields of artificial intelligence, complex systems, intelligent control and sensing, intelligent transportation systems, and intelligent vehicles. He has published over 100 SCI-indexed international journal articles and over 70 international conference papers as a first/corresponding author. He is a member of the Editorial Advisory Board for the *Transportation Research Part C: Emerging Technologies* and the Editorial Board for *Transport Reviews* and *ACTA Automatica*. He is an Associate Editor of the IEEE TRANSACTIONS ON INTELLIGENT TRANSPORTATION SYSTEMS and the IEEE TRANSACTIONS ON INTELLIGENT VEHICLES.

Youmin Zhang (Senior Member, IEEE) received the B.S., M.S., and Ph.D. degrees from Northwestern Polytechnical University, Xi'an, China, in 1983, 1986, and 1995, respectively. He is currently a Professor with the Department of Mechanical, Industrial and Aerospace Engineering, and the Concordia Institute of Aerospace Design and Innovation, Concordia University, Montreal, QC, Canada. His main research interests include fault detection and diagnosis (FDD), fault-tolerant control (FTC), fault-tolerant cooperative control (FTCC) of single and multiple unmanned aerial/space/ground/marine vehicles, smart grids, and applications of unmanned systems to forest fires, power lines, environment, natural resources and disasters monitoring, detection, and protection by combining with remote sensing techniques. He has authored eight books, over 550 journal articles, conference papers, and book chapters. He is a fellow of CSME and a Senior Member of AIAA. He is the President of the International Society of Intelligent Unmanned Systems and a member of the Technical Committee for several scientific societies. He has been an Editorial Board Member, Editor-in-Chief, Editor-at-Large, Editor, or Associate Editor of several international journals such as IEEE TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS, IET Cyber-systems and Robotics, Chinese Journal of Aeronautics, Journal of Systems Science and Complexity, Security and Safety, Unmanned Systems, Guidance, Navigation and Control, and *Journal of Intelligent and Robotic Systems*.