2019 IEEE 4th International Conference on Advanced
Robotics and Mechatronics (ICARM)

Fast Trajectory Planning for Off-Road Autonomous Driving with a Spatiotemporal Tunnel and Numerical Optimal Control Approach *

Bai Li, and Youmin Zhang

***Abstract*—This paper proposes a unified and fast trajectory planning method for an autonomous vehicle with avoidance of static obstacles in the off-road scenarios. Nominally, an optimal control problem should be used to cover generic off-road planning cases. However, the scale of the formulated optimal control problem is intractable, because the ego vehicle has to avoid collision with every obstacle at every moment. To address this issue, we aim to discard the redundant collision-avoidance constraints, because the vehicle may not really have chances to collide with all of the obstacles at every moment. Concretely, with the help of a reference trajectory derived by hybrid A* algorithm, we develop a spatiotemporal tunnel-based strategy such that the ego vehicle is restricted to move within a sequence of tunnels during different periods of the entire dynamic process. Through this, the collision-avoidance constraints are replaced by the within-tunnel constraints, thereby making the scale of the optimal control problem completely irrelevant to the complexity of the environment.**

I. Introduction

Autonomous vehicle technologies are bringing about positive changes in improving driving safety, enhancing traffic capacity, and reducing emissions [1]. In an autonomous driving system, the trajectory planning module is responsible for generating a spatiotemporal curve that is kinematically feasible for the vehicle, comfortable for the passengers, and collision-free from the sensored obstacles [2].

Trajectory planning algorithms for the on-road and off-road scenarios are quite different. Compared with on-road planning, off-road planning is more challenging because the planner should (i) support reverse driving motions, and (ii) deal with intricate obstacles. These two factors make the predominant on-road trajectory planners not applicable to the off-road cases. This paper deals with off-road trajectory planning for an autonomous vehicle.

The existing off-road trajectory planners primarily originate from robotics, and are classified into two categories: search-and-sample-based, and optimization-based methods [3]. A search-and-sample-based planner first abstracts the continuous state space as a graph containing finite numbers of nodes, and then searches for a satisfactory connection from the initial configuration towards the goal. Occupancy grids, random points, and state lattice are typical presentations of a graph. Optimization-based methods regard the trajectory planning task as an optimal control problem, which is about minimizing a predefined cost function subject to constraints [4–7]. Compared with the search-and-sample-based planners, formulating an optimal control problem is advantageous because (i) the continuous state space needs not discretized; and (ii) the trajectory is directly planned without path velocity decomposition. The numerical solution to an optimal control problem, nonetheless, is usually derived slowly, which limits the online applications of the optimization-based planners.

To make an optimization-based planner run fast, efforts have been made in three ways. The first way is to use the rough trajectory derived by a search-and-sample-based algorithm as the initial guess for warmly starting the numerical solution process of the optimal control problem [8]. The second way is to incorporate optimization into the sampler of a search-and-sample-based algorithm [9]. The third way is to simplify the optimal control problem formulation [10,11]. In this work, we aim to facilitate the online computation via a combination of the first and third way. Concretely, a rough trajectory is obtained through attaching a velocity along a rough path planned by hybrid A* algorithm. With that rough trajectory, we create a local neighborhood in the state space such that the complicated collision-avoidance constraints are converted into a sequence of within-tunnel restrictions. Refs. [10] and [12] held similar ideas of this work in part, and the contributions of this paper lie in that (i) trajectories instead of paths are planned for generic off-road cases; (ii) the derived trajectories are sufficiently smooth after a well-defined optimization process; and (iii) the efforts to simplify the optimal control problem does not involve extra offline preparation steps.

In the rest of this paper, Section II defines the trajectory planning problem in its nominal form. Section III introduces our real-time trajectory planning methodology. Simulation results are reported in Section IV. Finally, the conclusions are drawn in Section V.

II. Nominal Problem Statement

This section nominally defines the autonomous trajectory planning scheme as an optimal control problem, which is about minimizing the process completion time, subject to the kinematic constraints, collision-avoidance constraints, and two-point boundary conditions.

A. Vehicle Kinematics

Since the vehicle usually runs at low speeds in the off-road scenarios, the tire-sideslip effect is slim. Therefore, the bicycle model is sufficient to describe the vehicle kinematics [13,14]:

* Research supported by the National Natural Science Foundation of China under Grants 61573282 and 61833013, and the Natural Sciences and Engineering Research Council of Canada.

Bai Li is with the R&D Center of Automated Driving, JD Inc., Beijing 100176, China (e-mail: libai@zju.edu.cn).

Youmin Zhang is with the Department of Mechanical, Industrial and Aerospace Engineering, Concordia Institute of Aerospace Design and Innovation, Concordia University, Montreal, Canada (phone: 514-848-2424 ext. 5225; fax: 514-848-3175; e-mail: ymzhang@encs.concordia.ca).

978-1-7281-0064-7/19/$31.00 ©2019 IEEE

$$\frac{\mathrm{d}}{\mathrm{d}t}\begin{bmatrix} x(t) \\ y(t) \\ v(t) \\ \phi(t) \\ \theta(t) \end{bmatrix} = \begin{bmatrix} v(t)\cdot\cos\theta(t) \\ v(t)\cdot\sin\theta(t) \\ a(t) \\ \omega(t) \\ v(t)\cdot\tan\phi(t)/\mathrm{L_W} \end{bmatrix}, \ t\in[0,t_\mathrm{f}]. \qquad (1)$$

Herein, t is the time index, (x, y) refers to the mid-point of rear wheel axis (see point P in Fig. 1), θ refers to the orientation angle, v refers to the velocity of P, a refers to the corresponding acceleration, ϕ refers to the steering angle of the front wheels, ω refers to the corresponding angular velocity, and $\mathrm{L_W}$ denotes the wheelbase length. In addition, $\mathrm{L_F}$ is the front overhang length, $\mathrm{L_R}$ is the rear overhang length, and $\mathrm{L_B}$ is the vehicle width. Several boundaries are imposed to restrict the movement of the vehicle during the entire process $t\in[0,t_\mathrm{f}]$:

$$|a(t)|\le \mathrm{a_{max}}, \qquad (2a)$$

$$|v(t)|\le \mathrm{v_{max}}, \qquad (2b)$$

$$|\phi(t)|\le \Phi_{\mathrm{max}}, \qquad (2c)$$

$$|\omega(t)|\le \Omega_{\mathrm{max}}, \qquad (2d)$$

where $\mathrm{a_{max}}$, $\mathrm{v_{max}}$, Φ_{max}, and Ω_{max} represent the upper bounds of the corresponding variables, respectively.

B. Collision Avoidance Constraints

Suppose that the vehicle moves in a 2D Cartesian frame, and the obstacles in the environment are collected in a point set $\Theta=\left\{(\mathrm{obs\_x}_i,\mathrm{obs\_y}_i)\middle|i=1,2,...,\mathrm{N_{obs}}\right\}$, wherein $\mathrm{N_{obs}}$ denotes the number of obstacle points. Suppose that the vehicle is rectangular, and the four vertexes are A, B, C, and D (Fig. 1). The locations of $A(t)$, $B(t)$, $C(t)$, and $D(t)$ can be uniformly determined according to $x(t)$, $y(t)$, and $\theta(t)$. Nominally, collisions are avoided if the following restrictions are imposed:

$$\text{Point Q is out of } A(t)B(t)C(t)D(t),\ \forall \mathrm{Q}\in\Theta,\ \forall t\in[0,t_\mathrm{f}]. \qquad (3)$$

C. Two-Point Boundary Conditions

Boundary conditions consist of the configuration specifications at the initial and terminal moments. Concretely we have

$$\left[x(0),y(0),\theta(0),v(0),\phi(0),a(0),\omega(0)\right]= \left[\mathrm{x_0},\mathrm{y_0},\theta_0,\mathrm{v_0},\phi_0,\mathrm{a_0},\omega_0\right], \qquad (4a)$$

$$\left[x(t_\mathrm{f}),y(t_\mathrm{f}),\theta(t_\mathrm{f}),v(t_\mathrm{f}),\phi(t_\mathrm{f}),a(t_\mathrm{f}),\omega(t_\mathrm{f})\right]= \left[\mathrm{x_f},\mathrm{y_f},\theta_\mathrm{f},\mathrm{v_f},\phi_\mathrm{f},\mathrm{a_f},\omega_\mathrm{f}\right], \qquad (4b)$$

where $\mathrm{x_0}$, $\mathrm{y_0}$, θ_0, $\mathrm{v_0}$, ϕ_0, $\mathrm{a_0}$, ω_0, $\mathrm{x_f}$, $\mathrm{y_f}$, θ_f, $\mathrm{v_f}$, ϕ_f, $\mathrm{a_f}$, and ω_f are parameters which determine the starting and terminal configurations.

D. Overall Problem Formulation

The vehicle is expected to complete the movement subject to minimum time, thus t_f is set as the minimization objective of the problem.

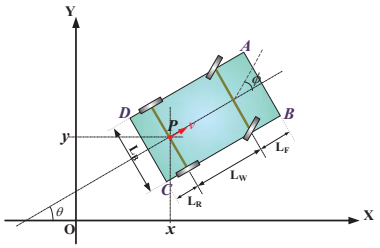

Fig. 1. Parametric notations related to vehicle shape and kinematics.

A general autonomous trajectory planning problem can be written as the following minimum-time optimal control problem.

$$\begin{aligned} &\min\ t_\mathrm{f}, \\ &\text{s.t. Kinematic constraints (1), (2);} \\ &\quad\text{Collision-avoidance constraints (3);} \\ &\quad\text{Initial and terminal conditions (4).} \end{aligned} \qquad (5)$$

Remark 2.1: The unknown variables of (5) include $x(t)$, $y(t)$, $\theta(t)$, $v(t)$, $a(t)$, $\omega(t)$, $\phi(t)$, and t_f.

Remark 2.2: The primary difficulties in (5) originate from the nonlinearity and dimensionality of the collision-avoidance constraints. Particularly, the scale of the obstacle points (i.e. $\mathrm{N_{obs}}$) has impacts on the complexity of the formulated optimal control problem.

III. Tunnel-Based Solution Facilitation Strategy

A. Motivations

According to (3), the ego vehicle needs to avoid collisions with *all* of the existing obstacle points at *every* moment during $[0,t_\mathrm{f}]$. However, the vehicle may not really have chances to collide with all of them at every moment. As depicted in Fig. 2, suppose a preliminary trajectory generated by a rough planner is available, then an optimization-based planner can find a topologically homotopic optimum if the preliminary trajectory serves as the initial guess [8]. This indicates that one can find a neighborhood of the preliminary trajectory in which the optimized trajectory stays. If so, the obstacles out of that neighborhood have no chance to collide with the ego vehicle. In Fig. 2, obstacle C lies out of the neighborhood, thus can be safely ignored. Besides that, the ego vehicle does not really have changes to collide with all of the rest obstacles at every moment. For example, when the vehicle approaches the goal, it is quite distant from obstacles A and F, thus the corresponding collision-avoidance constraints can be safely discarded when t approaches t_f. If we further adjust the neighborhood to exclude all the surrounding obstacle, a "green passage" is built. Through this, all of the collision-avoidance restrictions can be replaced by requesting the vehicle to travel in specified tunnels during specified time intervals. This transfer is beneficial because the complexity of the environment no longer affects the scale of the optimal control problem.

To conclude, our proposed solution facilitation strategy primarily consists of three steps, namely (i) deriving a

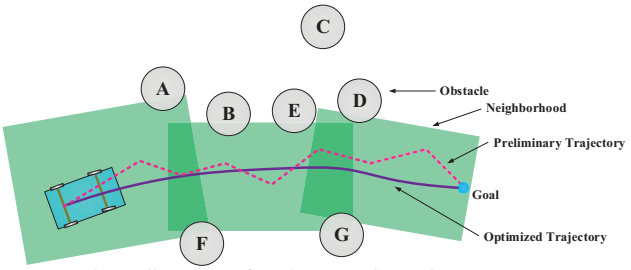

Fig. 2. Illustration of spatiotemporal tunnel pavement strategy.

preliminary trajectory via rough planning; (ii) re-formulating a spatiotemporal tunnel-based optimal control problem; and (iii) applying the preliminary trajectory as the initial guess to solve the re-formulated optimal control problem. Before all these steps, a scenario conversion step is developed as a pre-processing step. Detailed principles are introduced as follows.

B. Step 0: Scenario Conversion

According to (3), each obstacle point should locate out of the rectangular region *ABCD*. Such constraints, when analytically written as in [13], are highly non-convex, thus being difficult to handle. A common practice is using two discs to cover the rectangular vehicle body (Fig. 3). Suppose the two circle centers are denoted as $P_{\mathrm{f}}=(x_{\mathrm{f}},y_{\mathrm{f}})$ and $P_{\mathrm{r}}=(x_{\mathrm{r}},y_{\mathrm{r}})$. They are two quartile points along the vehicle's principal axis because the two circles evenly cover *ABCD*:

$$\begin{aligned}
x_{\mathrm{f}}(t)&=x(t)+\frac{1}{4}(3\mathrm{L_W}+3\mathrm{L_F}-\mathrm{L_R})\cdot\cos\theta(t),\\
y_{\mathrm{f}}(t)&=y(t)+\frac{1}{4}(3\mathrm{L_W}+3\mathrm{L_F}-\mathrm{L_R})\cdot\sin\theta(t),\\
x_{\mathrm{r}}(t)&=x(t)+\frac{1}{4}(\mathrm{L_W}+\mathrm{L_F}-3\mathrm{L_R})\cdot\cos\theta(t),\\
y_{\mathrm{r}}(t)&=y(t)+\frac{1}{4}(\mathrm{L_W}+\mathrm{L_F}-3\mathrm{L_R})\cdot\sin\theta(t).
\end{aligned}\tag{6a}$$

Denoting $\mathrm{R_C}$ as the radius of either circle, we have

$$\mathrm{R_C}=\frac{1}{2}\sqrt{(\frac{\mathrm{L_R}+\mathrm{L_W}+\mathrm{L_F}}{2})^2+(\mathrm{L_B})^2}.\tag{6b}$$

With the two discs to replace the rectangle *ABCD*, the collision-avoidance constraints (3) are formulated as

$$\begin{aligned}
&\left(x_{\mathrm{f}}(t)-\mathrm{x_Q}\right)^2+\left(y_{\mathrm{f}}(t)-\mathrm{y_Q}\right)^2\geq\mathrm{R_C}^2,\\
&\left(x_{\mathrm{r}}(t)-\mathrm{x_Q}\right)^2+\left(y_{\mathrm{r}}(t)-\mathrm{y_Q}\right)^2\geq\mathrm{R_C}^2,\\
&\forall\mathrm{Q}=(\mathrm{x_Q},\mathrm{y_Q})\in\Theta,\ \forall t\in[0,t_{\mathrm{f}}].
\end{aligned}\tag{7}$$

Eq. (7) requires that each obstacle point should keep at least a distance of $\mathrm{R_C}$ away from P_{f} and P_{r}. If one shrinks the vehicle body as the two mass points P_{f} and P_{r}, and dilates each obstacle point as a circle with the radius of $\mathrm{R_C}$, (7) still holds true. This means, such conversion of the scenario and vehicle shape is equivalent to the original one. Once converted, the situation can be regarded as two mass points with fixed relative distance traveling in a new digital map (Fig. 4). Later we will refer to this new map as *dilated map*.

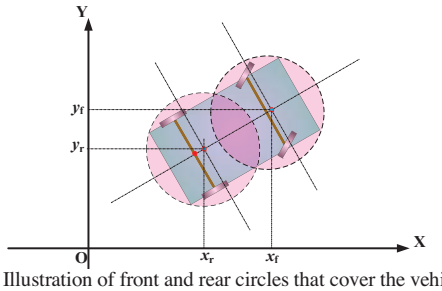

Fig. 3. Illustration of front and rear circles that cover the vehicle body.

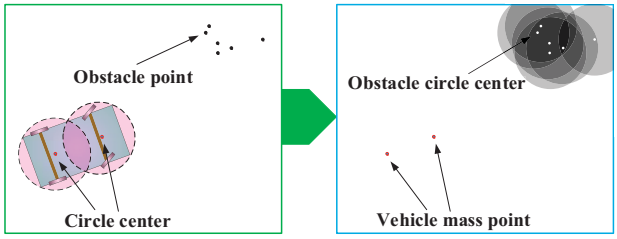

Fig. 4. Illustration of scenario conversion.

Readers may wonder why we do such an equivalent conversion. Section III.D shows that the collision-avoidance constraints we re-formulate are quite easy if the vehicle body is shrunk to two mass points.

C. Step 1: Reference Trajectory Generation

The aforementioned preliminary trajectory is formally introduced as the *reference trajectory*, which serves two purposes: (i) it helps to establish the tunnel; and (ii) it is used as the initial guess when numerically solving the formulated optimal control problem.

The reference trajectory should be planned fast because it is just roughly planned for future usage. Besides that, the reference trajectory should be kinematically feasible or at least near-feasible. Particularly in the off-road scenarios we concern, a vehicle may need to change the driving direction, thus the reference trajectory should support intermediate cusps. The hybrid A* algorithm [12], in association with some post-processing efforts, meets the aforementioned demands well.

Since the hybrid A* algorithm only plans a path, post-processing efforts are needed to convert the derived path to trajectory. Suppose that the derived path is represented in the form of a set $\Lambda=\left\{(\mathrm{x}_i,\mathrm{y}_i,\theta_i)\middle|i=1,2,...\right\}$, which contains a sequence of waypoints. In Λ, each element $(\mathrm{x}_i,\mathrm{y}_i,\theta_i)$ records the location of point *P* (i.e. the mid-point of rear wheel axis) as well as the vehicle's orientation angle. Suppose the path contains several cusps (i.e. direction-change points with $v=0$), and they divide the whole path into non-overlapping segments. The vehicle is expected to travel fast in each segment. With a time-optimal demand, the velocity is determined by analytically solving a one-dimensional minimal-time optimal control problem with bounded $a(t)$ and $v(t)$. With the computed $v(t)$ in each segment, one can estimate t_{f} (denoted as $\bar{t}_{\mathrm{f}}$ later) and the vehicle's configuration during the entire time domain $\left[0,\bar{t}_{\mathrm{f}}\right]$.

Since the vehicle is regarded as two fixed-distance mass points, the trajectory of point P should be converted into the trajectories of P_f and P_r. Eq. (6a) can be used to make the conversion. Suppose the obtained trajectories of P_f and P_r are denoted as $Traj_f$ and $Traj_r$, respectively.

D. Step 2: Trajectory Planning Problem Re-Formulation

This subsection aims to establish an easier version of (5). This is achieved by simplifying the collision-avoidance constraints (3) with the help of $Traj_f$ and $Traj_r$.

Let us focus on the movement of P_f at first. Nominally P_f should avoid collision with any of the obstacle points in the dilated map, which renders a large scale of collision-avoidance constraints. However, many parts of all the constraints are useless because P_f only has risks to collide with close-range obstacles at each instant. With the reference trajectory $Traj_f$ at hand, we can judge which obstacles are close to the mass point P_f at each instant. The detailed procedures are presented as follows.

First, let us sample as many as (N_R+1) waypoints in $Traj_f$ evenly in time. In other words, the locations of $P_f=(x_f(t),y_f(t))$ at $t=\frac{\bar{t}_f}{N_R}\cdot i$, $(i=0,1,...,N_R)$ are extracted from $Traj_f$ (see the example illustrated in Fig. 5). These points are referred to as *representative points* later. Through connecting the adjacent representative points, we can get as many as N_R line segments.

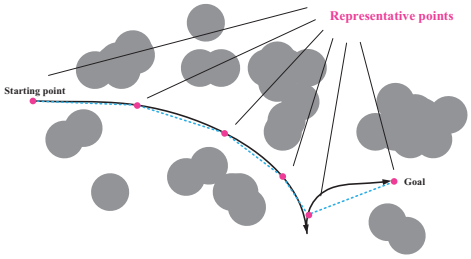

Fig. 5. Representative points in dilated map ($N_R = 5$).

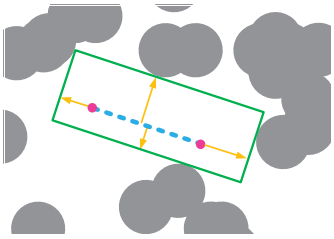

Fig. 6. Principle to generate representative rectangle.

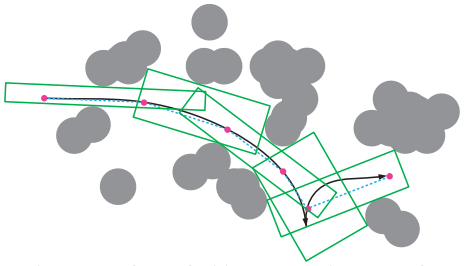

Fig. 7. Travel tunnel with representative rectangles.

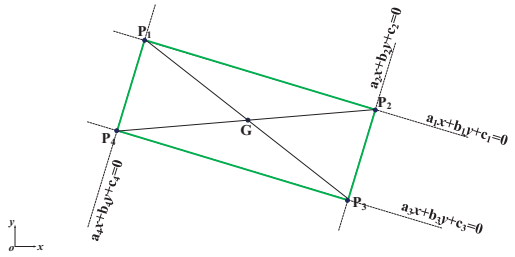

Fig. 8. Illustration of within-rectangle constraint formulation.

Second, a *representative rectangle* should be generated in association of each line segment. The generation principle is briefly presented here. Originally, the concerned line segment is regarded as a zero-width rectangle. With the line segment regarded as the skeleton (Fig. 6), we can incrementally expand the rectangle in the four directions by Δs each time. Once a trial of expansion in one direction is done, we check if this expansion trial would cause collisions with the obstacles in the dilated map. If no collision occurs, then the current expansion trial is approved; otherwise it is rejected and future expansion trials in this direction will not be considered any longer. We also introduce a parameter $L_{max\_expansion}$, which stands for the maximum allowable expansion length in each of the four directions. In this way, we can get as many as N_R representative rectangles for $Traj_f$ at a gross. As can be seen from Fig. 7, these representative rectangles fully cover the line segments, and form a series of tunnels from the starting point to the goal. With these representative rectangles, we can define the P_f-related collision-avoidance constraints as

$$\begin{aligned}&P_f(t) \text{ locates within the } i\text{th representative}\\ &\text{rectangle when } t\in[\frac{t_f}{N_R}\cdot(i-1),\ \frac{t_f}{N_R}\cdot i].\end{aligned} \tag{8}$$

Restricting a point S within an irregularly placed obstacle is quite easy. Let us focus on the constraint formulation within one representative rectangle (Fig. 8). Let us denote the four vertexes of the representative rectangle as P_1, P_2, P_3, and P_4. Point S should locate within the region surrounded by straight lines P_1P_2, P_2P_3, P_3P_4, and P_4P_1. Each straight line can be presented in the form of an equality. For example, line P_1P_2 is described as $a_1\cdot x+b_1\cdot y+c_1=0$ with

$$\begin{aligned}&a_1=y_2-y_1,\\ &b_1=x_1-x_2,\\ &c_1=x_2\cdot y_1-x_1\cdot y_2,\\ &P_1=(x_1,y_1),\ P_2=(x_2,y_2).\end{aligned} \tag{9}$$

Requiring that the point $S=(x_s,y_s)$ locates at one specific side of the straight line is described as $a_1\cdot x_s+b_1\cdot y_s+c_1>0$ or $a_1\cdot x_s+b_1\cdot y_s+c_1<0$. We need to make a choice between the two inequalities. Recall that point S and the rectangle's geometric center (denoted as G in Fig. 8) locate at the same side of the line. Thus we can simply formulate the constraint as

$$\begin{aligned}&g(S,a_1,b_1,c_1)<0,\ \text{where}\\ &\begin{cases}g=a_1\cdot x_s+b_1\cdot y_s+c_1, & \text{if } a_1\cdot x_G+b_1\cdot y_G+c_1<0\\ g=-a_1\cdot x_s-b_1\cdot y_s-c_1, & \text{otherwise}\end{cases}.\end{aligned} \tag{10}$$

The constraints in association with the rest three straight lines are formulated in the same way. Thus (8) can be expressed as the combination of four inequalities:

$$\begin{aligned} &g\left(P_{\mathrm{f}}(t),\mathrm{a}_1,\mathrm{b}_1,\mathrm{c}_1\right)<0,\ g\left(P_{\mathrm{f}}(t),\mathrm{a}_2,\mathrm{b}_2,\mathrm{c}_2\right)<0,\\ &g\left(P_{\mathrm{f}}(t),\mathrm{a}_3,\mathrm{b}_3,\mathrm{c}_3\right)<0,\ g\left(P_{\mathrm{f}}(t),\mathrm{a}_4,\mathrm{b}_4,\mathrm{c}_4\right)<0, \qquad (11)\\ &\forall t\in[\frac{t_{\mathrm{f}}}{\mathrm{N_R}}\cdot(i-1),\ \frac{t_{\mathrm{f}}}{\mathrm{N_R}}\cdot i]. \end{aligned}$$

In addition to P_{f}, P_{r} should avoid collisions as well. The constraint formulation principles are similar, thus are omitted. As a summary, there would be $\mathrm{N_R}$ representative rectangles for P_{f}, and $\mathrm{N_R}$ representative rectangles for P_{r}. There would be totally $8\mathrm{N_R}$ types of inequalities which prevent the vehicle from collisions with the obstacle points in the dilated map:

$$\begin{cases} g\left(P_{\mathrm{f}}(t),\mathrm{a}_{fi1},\mathrm{b}_{fi1},\mathrm{c}_{fi1}\right)<0\\ g\left(P_{\mathrm{f}}(t),\mathrm{a}_{fi2},\mathrm{b}_{fi2},\mathrm{c}_{fi2}\right)<0\\ g\left(P_{\mathrm{f}}(t),\mathrm{a}_{fi3},\mathrm{b}_{fi3},\mathrm{c}_{fi3}\right)<0\\ g\left(P_{\mathrm{f}}(t),\mathrm{a}_{fi4},\mathrm{b}_{fi4},\mathrm{c}_{fi4}\right)<0 \end{cases},\ \begin{cases} g\left(P_{\mathrm{r}}(t),\mathrm{a}_{ri1},\mathrm{b}_{ri1},\mathrm{c}_{ri1}\right)<0\\ g\left(P_{\mathrm{r}}(t),\mathrm{a}_{ri2},\mathrm{b}_{ri2},\mathrm{c}_{ri2}\right)<0\\ g\left(P_{\mathrm{r}}(t),\mathrm{a}_{ri3},\mathrm{b}_{ri3},\mathrm{c}_{ri3}\right)<0\\ g\left(P_{\mathrm{r}}(t),\mathrm{a}_{ri4},\mathrm{b}_{ri4},\mathrm{c}_{ri4}\right)<0 \end{cases}, \qquad (12)$$

$$\forall t\in[\frac{t_{\mathrm{f}}}{\mathrm{N_R}}\cdot(i-1),\ \frac{t_{\mathrm{f}}}{\mathrm{N_R}}\cdot i],\ i=1,...,\mathrm{N_R}.$$

Herein, $[\mathrm{a}_{fik},\mathrm{b}_{fik},\mathrm{c}_{fik}]$ $(k=1,2,3,4)$ denote the coefficients related to the ith representative rectangle for P_{f}. Similarly, $[\mathrm{a}_{rik},\mathrm{b}_{rik},\mathrm{c}_{rik}]$ $(k=1,2,3,4)$ denote the coefficients related to the ith representative rectangle for P_{r}.

Remark 3.1: Compared with (7) which requires P_{f} and P_{r} to avoid all of the obstacles at every moment, (12) restricts the two points to be within specific regions in different time intervals. Eq. (12) is efficient to reduce the unnecessary parts in (7).

Remark 3.2: The scale of (12) is determined only by $\mathrm{N_R}$, but not by the number of obstacle grids in the dilated map.

Remark 3.2 means that the complexity of the environment does not have any impact on the scale of the formulated trajectory planning problem.

As a conclusion of the whole subsection, a new trajectory planning problem is formulated:

$$\begin{aligned} &\min\ t_{\mathrm{f}},\\ &\text{s.t. Kinematic constraints (1), (2);}\\ &\quad\text{Collision-avoidance constraints (12);}\\ &\quad\text{Initial and terminal conditions (4).} \end{aligned} \qquad (13)$$

Remark 3.3: Compared with (5), the dimensionality and nonlinearity of (13) are reduced.

Remark 3.4: The solution to (13) may not be the optimal solution to the nominal problem (5) due to the following reasons: (i) the adopted hybrid A* algorithm may provide a sub-optimal topology for the generation of reference trajectory; and (ii) dividing the environment into rectangles sacrifices travelable areas (see the white regions out of the rectangles in Fig. 7).

Although there are potential risks of losing the optimality according to Remark 3.4, the solution to (13) are usually close to the optimal one.

E. Step 3: Optimal Control Problem Solution

In order to solve the re-formulated optimal control problem (13) numerically, the time domain $[0,t_{\mathrm{f}}]$ should be discretized so that (13) is converted into a nonlinear programming (NLP) problem [15]. In this work, we choose the first-order explicit Runge-Kutta method to do the discretization and adopt the Interior-Point Method (IPM) [16] for NLP solution. We utilize the aforementioned reference trajectory to warmly start the NLP optimization process. Finally, IPM outputs the optimized trajectory.

IV. Simulation Results

Simulations were performed in C++ and executed on an i5-7200U CPU with 8 GB RAM that runs at 2.50 × 2 GHz. Basic parametric settings are listed in Table I.

TABLE I. Parametric Settings regarding Model and Approach.

Parameter	Description	Setting
$\mathrm{L_F}$	Front hang of vehicle	0.55 m
$\mathrm{L_W}$	Wheelbase of vehicle	0.85 m
$\mathrm{L_R}$	Rear hang of vehicle	0.40 m
$\mathrm{L_B}$	Width of vehicle	0.80 m
$\mathrm{a_{max}}$	Bound of $\lvert a(t)\rvert$	Inf
$\mathrm{v_{max}}$	Bound of $\lvert v(t)\rvert$	1.0 m/s
Φ_{max}	Bound of $\lvert\phi(t)\rvert$	0.30 rad
Ω_{max}	Bound of $\lvert\omega(t)\rvert$	0.5 rad/s
Δs	Increment of step in representative rectangle enlargement algorithm	0.1 m
$\mathrm{L_{max\_expansion}}$	Maximum step length in representative rectangle enlargement algorithm	4.0 m
NE	Number of finite elements in using Runge-Kutta method for creating NLP	60

Recall that the tunnel-based strategy consists of three stages, namely the reference trajectory generation (Stage 1), optimal control problem re-formulation (Stage 2), and optimal control problem solution (Stage 3). We would investigate the time consumption of each stage.

A large number of randomly defined trajectory planning cases have been tested. Herein, the randomness is reflected by (i) randomly defined starting point $\left(x(0),y(0)\right)$ in an $80\ \mathrm{m}\times 80\ \mathrm{m}$ map; (ii) randomly defined ending point $\left(x(t_{\mathrm{f}}),y(t_{\mathrm{f}})\right)$ in the same map, which keeps a distance L from $\left(x(0),y(0)\right)$ $(\mathrm{L}\le 10)$; (iii) the starting and terminal orientations $\theta(0)$ and $\theta(t_{\mathrm{f}})$ are independently and randomly defined along the interval $[0,2\pi)$; and (iv) as many as $\mathrm{N_{obs}}$ obstacle points are independently and randomly set in the map, wherein $\mathrm{N_{obs}}\in[10,100]$ is a random number. 179,421 random cases have been tested, and none of the simulation tests crashed, which indicates the tunnel-based strategy is robust. The results are summarized in Table II.

TABLE II. TIME CONSUMPTIONS ON EACH STAGES.

KPI \ Stage #	Stage 1	Stage 2	Stage 3
Success rate	99.54%	100%	98.69%
Average CPU time for succeeded cases (sec)	0.0210	0.1024	0.0195
Average CPU time for failed cases (sec)	0.0222	-	0.1382
Average CPU time (sec)	0.0210	0.1024	0.0564
Maximum CPU time (sec)	2.2597	0.2623	3.2636
99th percentile CPU time (sec)	0.0656	0.1400	0.1671

With regard to Stage 1, it is notable in Table II that the hybrid A* algorithm can provide valid reference trajectories for most of the cases, while failed in a small portion of cases. The failures do not necessarily mean the incapability of the hybrid A* algorithm. Instead, the aforementioned way to generate random cases may render cases that do not have any feasible solutions at Stage 1. Regardless of a success or failure, the average CPU time spent at Stage 1 is nearly the same, which indicates the hybrid A* algorithm is stable. Stage 2 did not involve any failure, because no complicated calculation is done there. Commonly, the CPU time spent at Stage 2 is short if the environment is tiny, otherwise more incremental trials of expansions would be implemented, which costs time. In terms of Stage 3, the failed cases spent longer time than the succeeded cases. This is reasonable because the gradient-based solver (i.e. IPM) needs extra iteration steps to find if there are alternative search directions that would lead to feasibility in the solution space. As a whole, the tunnel-based strategy averagely spent less than 200 ms to provide an optimized trajectory, and the gross CPU time is no larger than 400 ms for 99% of the cases.

Before the end of this section, the optimized trajectory for a parking scheme is depicted in Fig. 9. A few features are observed: (i) the representative rectangles always tend to cover the reference trajectory to the largest extent; (ii) the reference trajectory is different from the optimum; and (iii) the reference trajectory usually differs from the optimized trajectory when the steering angle is non-zero. Feature (iii) indicates that the reference trajectory derived by the hybrid A* algorithm is not a feasible solution when the vehicle makes a turn. Inherently, this is because the Reeds-Shepp curves incorporated in the hybrid A* algorithm are not continuous-curvature.

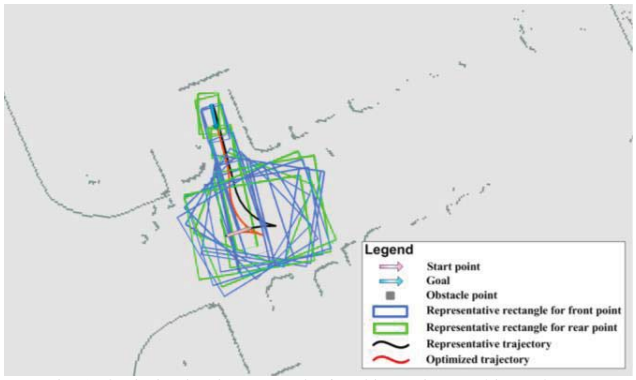

Fig. 9. Optimized trajectory and related items in a parking scheme.

V. CONCLUSIONS

This paper has introduced a fast trajectory planning method for generic off-road autonomous driving cases. The hybrid A* algorithm, the tunnel creation principle, as well as the IPM involved in this work may be replaced by alternatives if they take the desired functions. This indicates that our proposal is inherently an open framework rather than a specific algorithm. Comparisons will be conducted in the future.

REFERENCES

[1] 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 Transactions on Intelligent Vehicles*, vol. 1, no. 1, pp. 33–55, 2016.

[2] D. González, J. Pérez, V. Milanés, and F. Nashashibi, "A review of motion planning techniques for automated vehicles," *IEEE Transactions on Intelligent Transportation Systems*, vol. 17, no. 4, pp. 1135–1145, 2016.

[3] C. Liu, C. Y. Lin, and M. Tomizuka, "The convex feasible set algorithm for real time optimization in motion planning," *SIAM Journal on Control and Optimization*, vol. 56, no. 4, pp. 2712–2733, 2018.

[4] B. Li, K. Wang, and Z. Shao, "Time-optimal maneuver planning in automatic parallel parking using a simultaneous dynamic optimization approach," *IEEE Transactions on Intelligent Transportation Systems*, vol. 17, no. 11, pp. 3263–3274, 2016.

[5] K. Kondak and G. Hommel, "Computation of time optimal movements for autonomous parking of non-holonomic mobile platforms," In *Proc. 2001 IEEE International Conference on Robotics and Automation (ICRA)*, vol. 3, pp. 2698–2703, 2001.

[6] K. Bergman, and D. Axehill, "Combining homotopy methods and numerical optimal control to solve motion planning problems," In *Proc. 2018 IEEE Intelligent Vehicles Symposium (IV)*, pp. 347–354, 2018.

[7] 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. 2019 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 8557–8562, 2019.

[8] C. Rösmann, F. Hoffmann, and T. Bertram, "Integrated online trajectory planning and optimization in distinctive topologies," *Robotics and Autonomous Systems*, vol. 88, pp. 142–153, 2017.

[9] A. Perez, et al., "LQR-RRT*: Optimal sampling-based motion planning with automatically derived extension heuristics," In *Proc. 2012 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 2537–2542, 2012.

[10] H. Andreasson, et al., "Fast, continuous state path smoothing to improve navigation accuracy," In *Proc. 2015 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 662–669, 2015.

[11] X. Zhang, A. Liniger, A. Sakai, and F. Borrelli, "Autonomous parking using optimization-based collision avoidance," In *Proc. 2018 IEEE Conference on Decision and Control (CDC)*, pp. 4327–4332, 2018.

[12] D. Dolgov, S. Thrun, M. Montemerlo, and J. Diebel, "Path planning for autonomous vehicles in unknown semi-structured environments," *International Journal of Robotics Research*, vol. 29, no. 5, pp. 485–501, 2010.

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

[14] B. Li, Y. M. Zhang, and Z. Shao, "Spatio-temporal decomposition: a knowledge-based initialization strategy for parallel parking motion optimization," *Knowledge-Based Systems*, vol. 107, pp. 179–196, 2016.

[15] B. Li, H. Liu, D. Xiao, G. Yu, and Y. M. Zhang, "Centralized and optimal motion planning for large-scale AGV systems: A generic approach," *Advances in Engineering Software*, vol. 106, pp. 33–46, 2017.

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