Convergence-Guaranteed Trajectory Planning for a Class of Nonlinear Systems with Nonconvex State Constraints

Xinfu Liu

***Abstract*—In this paper, we study the problem of trajectory planning for a class of nonlinear systems with convex state/control constraints and nonconvex state constraints with concave constraint functions. This corresponds to a challenging nonconvex optimal control problem. We present how to convexify the nonlinear dynamics without any approximation via a combination of variable redefinition and relaxation. We then prove that the relaxation is exact by designing an appropriate objective function. This exact relaxation result enables us to further convexify the nonconvex state constraints simply by linearization. A a result, an algorithm is designed to iteratively solve the obtained convex optimization problems until convergence to get a solution of the original problem. A unique feature of the proposed approach is that the algorithm is proved to converge and it does not rely on any trust-region constraint. High performance of the algorithm is demonstrated by its application to trajectory planning of UAVs and autonomous cars with obstacle avoidance requirements.**

***Index Terms*—Trajectory planning, Relaxation, Convex optimization, Guidance and control of vehicles**

I. Introduction

The task of trajectory planning is to find a dynamically feasible and possibly optimal trajectory from a given initial state to a target state or region while satisfying all necessary constraints arising from physical limitations or operational safety. It is a core technology for a wide range of vehicles to execute certain missions, such as autonomous driving of cars, collision avoidance of UAVs, rendezvous and docking of space vehicles, etc. Trajectory planning based on optimal control is an important method widely used to find trajectories for various missions, mainly because it has the benefit of explicitly taking into account system dynamics and necessary state/control and terminal constraints for mission accuracy and success [1], [2], [3], [4]. The associated optimal control problems (OCPs) are generally solved as nonlinear programming (NLP) problems using various numerical methods [5], [6].

Autonomous and advanced missions require trajectory planning to be performed onboard in real time [7], [8], [9], [10], [11]. Nevertheless, this requirement is difficult to meet, because the associated OCPs are generally nonconvex and most nonconvex optimization problems are hard to solve in terms of long computation time and no guarantee of finding a solution [12]. This is also the difficulty that the well-known nonlinear model predictive control (NMPC) technique often faces, since the nonlinear dynamics and possible existence of nonlinear constraints will make the optimization problem at each sample instant nonconvex. Hence, it is still an open issue to realize real-time implementation of NMPC for systems with fast dynamics [13].

Significantly improving the reliability and efficiency of solving nonconvex OCPs is critical for autonomous systems to realize real-time trajectory planning. Apart from advances in computing hardware, there are two types of efforts for the objective, one is at the solver level and the other is at the problem level. At the solver level we can apply techniques, such as warm starting and structure exploiting, to customize numerical solvers so that the computation burden can be significantly reduced for general NLP problems [14] and convex quadratic or second-order cone programming (SOCP) problems [15], [16], [17], [18], [19]. At the problem level we make an effort to reformulate or transform the original problem into much easier problem(s) to achieve higher solution robustness and efficiency [20]. For instance, for a class of OCPs with linear systems and the only nonconvexity arising from control constraints, we can first relax the control constraints and then solve the relaxed problem to obtain a solution of the original problem [21]. This result was also extended to the case when linear state constraints are active over some finite time intervals [22]. An advantage of relaxing the control constraints is that we only need to solve a single convex problem which is guaranteed to be solved in polynomial time [12], [23]. This convergence-guaranteed feature has enabled a number of technological innovations in planetary soft landing [24], [25], [26] with flight tests conducted on a suborbital rocket [27], [28]. Practical applications are likely to be seen in future robotic or human lander missions to the Moon, Mars, or other solid-surface destinations, which require high landing accuracy [29]. For more complex problems such as those with nonlinear dynamics and/or nonlinear state constraints, we may use various techniques, including linearization, to approximate the original problem with a convex problem, and then iteratively solve the convex problem to get a solution [30], [31]. This method is attractive since the convex problem in each iteration can be very efficiently solved by interior point methods with guaranteed convergence. However, achieving convergence of the iterative solution procedure may still be a challenge [32].

In this paper, we study the trajectory planning problem for a special class of nonlinear systems with convex state/control constraints and nonconvex state constraints. The purpose of this paper is to seek techniques at the problem level to solve

X. Liu is with the School of Aerospace Engineering, Beijing Institute of Technology, Beijing 100081, China. E-mail: xliu@bit.edu.cn

the associated nonconvex OCP in real time with guaranteed convergence. Note that the nonlinear systems considered can include the motion of cars, mobile robots, UAVs, missiles, etc. The nonconvex state constraints are used to represent the collision or obstacle avoidance requirements that occur in many autonomous applications. In general, to solve the problem via convex optimization, nonlinear dynamics are linearized to get linear dynamics and other nonconvex constraints are linearized as well. Then, dynamically adjusting the radius of a trust region constraint can achieve convergence of the iterative solution procedure under some assumptions [31]. In this paper, we will convexify the nonlinear dynamics without any approximation and we will show how to ensure the validity of the covexification process. This is different from approximating the nonlinear dynamics with linear dynamics through linearization. A promising benefit of the proposed method will be seen in our ability to design a very efficient iterative algorithm with guaranteed convergence and no trust-region constraints to get a solution of the original problem.

This paper is organized as follows. The trajectory planning problem for the class of nonlinear systems is introduced in Sec. II. We convexify the nonlinear system dynamics and obtain a relaxation of the original problem in Sec. III, and how to ensure exactness of the relaxation is discussed in Sec. IV. Section V shows how to handle the nonconvex state constraints and presents an iterative algorithm with guaranteed convergence to get a solution of the original problem. Applications on collision-avoidance trajectory planning of UAVs and autonomous cars are shown in Sec. VI and we conclude this paper in Sec. VII.

II. Problem statement

In this paper, we consider the problem of trajectory planning for a class of nonlinear systems with the following system dynamics

$$\begin{aligned} \dot{x}(t) &= v(t)\cos\theta(t), \quad & x(t_0) &= x_0 \\ \dot{y}(t) &= v(t)\sin\theta(t), \quad & y(t_0) &= y_0 \\ \dot{\theta}(t) &= w(t), & \theta(t_0) &= \theta_0, \end{aligned} \tag{1}$$

where $(x(t), y(t))$ is the position in a Cartesian coordinate system, $\theta(t)$ is the heading angle measured counterclockwise from the x-axis, $v(t)$ is the velocity (time-varying or constant), $w(t)$ is the angular velocity, and (x_0, y_0, θ_0) is the initial value of the system state. The above system dynamics are commonly used to describe car-like vehicles [9], [33], mobile robots [34], [35], UAVs [36], and missiles [37], [38]. Note that the explicit time dependence of variables will be dropped in the rest of this paper for notation convenience.

In the following, some constraints will be introduced and finally we will get an optimal control problem related to trajectory planning for the class of nonlinear systems in (1).

A. Constraints

First, the control input w in (1) is usually bounded as

$$|w| \le w_{\max}, \tag{2}$$

where $w_{\max}$ is the maximum allowed angular velocity. It is a convex control constraint. For an autonomous car, the bound $w_{\max}$ can be given by [9]

$$w_{\max} = \frac{v}{l}\tan(\delta_{\max}),$$

where l is the distance between the front wheel and the rear wheel, and $\delta_{\max}$ represents the maximum allowed steering angle at the velocity v.

Second, convex state constraints are considered as follows

$$g_j^c(x, y) \le 0,\ j = 1, \ldots, n^c, \tag{3}$$

where g_j^c is a linear function or a convex quadratic function (the superscript 'c' represents '*convex*'), and n^c is the number of constraints. Note that when g_j^c is a convex quadratic function, $g_j^c \le 0$ is second-order cone representable [39]. The constraints in Eq. (3) might be from boundary limits of a given arena or from a requirement of moving in specific corridors.

Third, we also consider nonconvex state constraints in the following form

$$g_j^{nc}(x, y) \le 0,\ j = 1, \ldots, n^{nc}, \tag{4}$$

where g_j^{nc} is C^2 and *concave* (the superscript '*nc*' represents '*nonconvex*'), and n^{nc} is the number of constraints. This constraint can typically arise from requirements of avoiding obstacles, staying away from no-fly zones, and keeping a safe distance away from neighboring vehicles. For instance, if it is required to avoid an elliptical obstacle, the constraint is

$$1 - \left(\frac{x - x_{cj}}{a_j}\right)^2 - \left(\frac{y - y_{cj}}{b_j}\right)^2 \le 0, \tag{5}$$

where (x_{cj}, y_{cj}) is the center of the j-th ellipse and a_j (b_j) are the semimajor (semiminor) axes. The left side in Eq. (5) is a concave function of C^2.

Finally, we consider the following convex terminal constraints

$$\phi_j(x(t_f), y(t_f), \theta(t_f)) \le 0,\ j = 1, \ldots, n^t, \tag{6}$$

where t_f is a given final time, n^t is the number of terminal constraints, and ϕ_j is assumed to be affine in its arguments. Requiring a vehicle to reach a target position (x_f, y_f) at t_f can be contained in the above constraints.

B. Optimal control problem

Finding a trajectory satisfying the system dynamics (1), path constraints (2)-(4), and terminal constraints (6) can be formulated as the following OCP

$$\begin{aligned} \text{Problem } \mathcal{O}: \ \min_{w}\ & J = \varphi(\bar{z}(t_f), t_f) + \int_{t_0}^{t_f} \ell(\bar{z}, w, t)\mathrm{d}t \\ \text{s.t. }\ & \dot{x} = v\cos\theta,\ x(t_0) = x_0 \\ & \dot{y} = v\sin\theta,\ y(t_0) = y_0 \\ & \dot{\theta} = w,\ \theta(t_0) = \theta_0 \\ & |w| \le w_{\max} \\ & g^c(x, y) \le 0 \\ & g^{nc}(x, y) \le 0 \\ & \phi(\bar{z}(t_f)) \le 0, \end{aligned} \tag{7}$$

where $\bar{z}(t) = [x(t)\ \ y(t)\ \ \theta(t)]^T \in \mathbb{R}^3$ is the system state, $\varphi : \mathbb{R}^3 \times \mathbb{R} \to \mathbb{R}$ is a terminal cost, and $\ell : \mathbb{R}^3 \times \mathbb{R} \times \mathbb{R} \to \mathbb{R}$ is an integrand to form a running cost.

Note that at this moment a specific form of the objective function is not given. This is because we later need to carefully design it so that a relaxation technique introduced in the next section is *exact*. For now, let us simply consider that it is a convex function.

For Problem $\mathcal{O}$, we make the following two assumptions

Assumption 1: The heading angle satisfies $|\theta| < \frac{\pi}{2}$.

Assumption 2: For the control constraint and the $n^c + n^{nc}$ state constraints in Problem $\mathcal{O}$, at most one of them is active in any finite interval.

Remark 1:

(a) Assumption 1 means that the work in this paper is limited to missions in which a vehicle does not move backward. This assumption is often true, while in cases where it is not true, we may adjust the x-axis of the coordinate frame so that the heading angle defined in the new coordinate frame satisfies the assumption. Nevertheless, it should be pointed out that for missions in which having $|\theta| \geq \pi/2$ is inevitable, the corresponding problem is generally much more challenging to solve and this is considered out of the scope of this paper.

(b) Assumption 2 means that for the $n^c+n^{nc}+1$ constraints, including one control constraint and n^c+n^{nc} state constraints, any two of them can not be simultaneously active in any finite interval. Note that this is a very mild assumption. Physically speaking, a vehicle driven by a control input $w_{\max}$ or $-w_{\max}$ generally does not happen to keep moving along a constraint boundary defined by any active state constraint. In other words, moving along a state constraint boundary generally does not correspond to a control $w_{\max}$ or $-w_{\max}$. In addition, it is generally true that there is at most one active state constraint in a finite interval, unless some of the state constraints are redundant. Nevertheless, redundant state constraints should be avoided when we formulate Problem $\mathcal{O}$.

Next, we need to solve Problem $\mathcal{O}$. It is nonconvex due to the nonlinear system dynamics (1) and the nonconvex state constraints (4). Hence, it is difficult to solve the problem, especially when our goal in this paper is to design an algorithm to solve it with guaranteed convergence and high efficiency. The rest of this paper will be devoted to developing appropriate techniques to effectively convexify the problem and achieve the goal.

III. Convexification of nonlinear dynamics

In this section, we present how to convexify the nonlinear system dynamics (1) into linear dynamics and the convexification process may generate additional constraints. We advocate that in the convexfication process (certain) nonlinearity inherent in the original nonlinear system dynamics should be preserved if possible, and the preserved nonlinearity can lie in the additional nonlinear constraints. Such a strategy will play a significant role in helping us design a convergence-guaranteed algorithm to solve the nonconvex Problem $\mathcal{O}$.

A. Preserving nonlinearity in system dynamics

For the dynamics (1), we define two new variables as follows

$$\theta_c := \cos\theta \tag{8}$$

$$\theta_s := \sin\theta. \tag{9}$$

Then, the original nonlinear dynamics (1) become

$$\dot{x} = v\,\theta_c \tag{10}$$

$$\dot{y} = v\,\theta_s \tag{11}$$

$$\frac{\dot{\theta_s}}{\theta_c} = w, \tag{12}$$

where $\dot{\theta} = \dot{\theta_s}/\theta_c$ is used. It is noticed that Eqs. (10)-(11) are now linear, but Eq. (12) is still nonlinear. By introducing a new variable θ_d and letting $\dot{\theta_s}$ equal to θ_d, we can equivalently express Eq. (12) as

$$\dot{\theta_s} = \theta_d \tag{13}$$

$$\frac{\theta_d}{\theta_c} = w. \tag{14}$$

Note that Eq. (13) is linear, whereas Eq. (14) is a nonlinear constraint. When Eq. (14) is combined with the original control constraint in Eq. (2), it yields

$$\left|\frac{\theta_d}{\theta_c}\right| \leq w_{\max}. \tag{15}$$

Under Assumption 1, $\theta_c = \cos\theta > 0$ and thus Eq. (15) can be further rewritten as the following convex constraints

$$\theta_d \geq -w_{\max}\,\theta_c \tag{16}$$

$$\theta_d \leq w_{\max}\,\theta_c. \tag{17}$$

It is worth noting that in the absence of Assumption 1, Eq. (15) is a nonconvex constraint and it will be more challenging to find an *effective* method to convexity it.

Though the definitions in Eqs. (8)-(9) and the inclusion of the new variable θ_d are effective to obtain the linear dynamics in Eqs. (10)-(11), and (13), the following new constraint should be imposed

$$(\theta_c)^2 + (\theta_s)^2 = 1. \tag{18}$$

Now, it can be seen from the above conversion process that the original dynamics in Eq. (1) plus the control constraint in Eq. (2) are converted into the new linear dynamics in Eqs. (10)-(11), (13) plus the constraints in Eqs. (16)-(18). Note that in the new dynamics x, y, and θ_s are the state variables, while θ_c and θ_d are viewed as the control inputs. The original control input w disappears, but it can be recovered by $w = \theta_d/\theta_c$ [see Eq. (14)].

For the convenience of notation, we denote

$$\begin{aligned} z &= [z_1\ z_2\ z_3]^T := [x\ y\ \theta_s]^T \\ u &= [u_1\ u_2]^T := [\theta_c\ \theta_d]^T. \end{aligned} \tag{19}$$

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TAES.2021.3131140, IEEE Transactions on Aerospace and Electronic Systems

Then, Problem $\mathcal{O}$ is transformed into the following new OCP

$$\begin{aligned}\text{Problem } \mathcal{E}: \quad \min_{u} \quad & J = \varphi(z(t_f), t_f) + \int_{t_0}^{t_f} \ell(z,u,t)\mathrm{d}t \\ \text{s.t.} \quad & \dot{z} = A\,z + B\,u,\ z(t_0) = z_0 \\ & u_2 \geq -w_{\max}\,u_1 \\ & u_2 \leq w_{\max}\,u_1 \\ & (u_1)^2 + (z_3)^2 = 1 \\ & g^c(x,y) \leq 0 \\ & g^{nc}(x,y) \leq 0 \\ & \phi(z(t_f)) \leq 0,\end{aligned} \tag{20}$$

where

$$A = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & v \\ 0 & 0 & 0 \end{bmatrix}, \quad B = \begin{bmatrix} v & 0 \\ 0 & 0 \\ 0 & 1 \end{bmatrix}. \tag{21}$$

Note that in the state constraints of Problem $\mathcal{E}$ we keep using x and y, instead of z_1 and z_2, without causing any confusion [cf. Eq. (19)]. It is to highlight that the state constraints are only dependent on the position coordinates x and y.

Obviously, Problem $\mathcal{E}$ is equivalent to Problem $\mathcal{O}$, since no approximation is used when the original nonlinear dynamics are convexified. This result is stated in the following lemma

Lemma 1: Under Assumption 1, Problem $\mathcal{E}$ is equivalent to Problem $\mathcal{O}$.

A notable difference between Problem $\mathcal{E}$ and Problem $\mathcal{O}$ is that the dynamics in Problem $\mathcal{E}$ are linear. Nevertheless, the nonlinearity in the original dynamics of Problem $\mathcal{O}$ are not removed. It is actually preserved because the newly generated constraint $(u_1)^2 + (z_3)^2 = 1$ in Problem $\mathcal{E}$ is nonlinear. Note that this is right the idea of preserving nonlinearity inherent in the original dynamics. Specifically, it involves transferring (certain) nonlinearity in the original dynamics into nonlinearity in the constraints. In general, though the newly generated constraints are nonlinear and nonconvex, they are easier to be effectively convexified than the direct convexification of the nonlinear dynamics.

In the following subsection, we will discuss how to convexify the mixed state and control constraint $(u_1)^2 + (z_3)^2 = 1$ in Problem $\mathcal{E}$.

B. Relaxing the mixed state and control constraint

The mixed state and control constraint in Problem $\mathcal{E}$ is nonconvex. We propose to relax it into the following relaxed constraint

$$(u_1)^2 + (z_3)^2 \leq 1, \tag{22}$$

which is convex. Note that relaxing the nonconvex constraint does not lose any nonlinearity. That is to say, any solution that is feasible to the mixed state and control constraint is also feasible to the above constraint.

After the mixed state and control constraint in Problem $\mathcal{E}$ is relaxed, we get the following relaxed problem denoted as

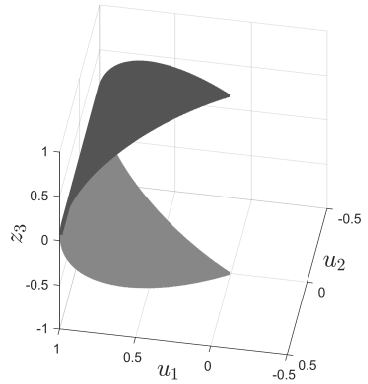

Fig. 1: The feasible set defined by the constraints $u_2 \geq -w_{\max}\,u_1$, $u_2 \leq w_{\max}\,u_1$, and $(u_1)^2 + (z_3)^2 = 1$ in Problem $\mathcal{E}$.

Problem $\mathcal{R}$ (note that Problem $\mathcal{R}$ is also called as a *relaxation* of Problem $\mathcal{E}$)

$$\begin{aligned}\text{Problem } \mathcal{R}: \quad \min_{u} \quad & J = \varphi(z(t_f), t_f) + \int_{t_0}^{t_f} \ell(z,u,t)\mathrm{d}t \\ \text{s.t.} \quad & \dot{z} = A\,z + B\,u,\ z(t_0) = z_0 \\ & u_2 \geq -w_{\max}\,u_1 \\ & u_2 \leq w_{\max}\,u_1 \\ & (u_1)^2 + (z_3)^2 \leq 1 \\ & g^c(x,y) \leq 0 \\ & g^{nc}(x,y) \leq 0 \\ & \phi(z(t_f)) \leq 0.\end{aligned} \tag{23}$$

From Problem $\mathcal{E}$ to Problem $\mathcal{R}$, the feasible set defined by the pure control constraints and the mixed state and control constraint is enlarged, as visually illustrated in Figs. 1-2. Obviously, Problem $\mathcal{R}$ contains all possible solutions of Problem $\mathcal{E}$, while a benefit of the relaxation technique is that the set shown in Fig. 2 is convex, which makes Problem $\mathcal{R}$ easier to solve than Problem $\mathcal{E}$. If an optimal solution to Problem $\mathcal{R}$ lies on the curved surface of the set in Fig. 2, or mathematically the relaxed constraint (22) is active, this solution will be feasible to Problem $\mathcal{E}$. In addition, the feasible set of Problem $\mathcal{E}$ is a subset of that of Problem $\mathcal{R}$ and the problem costs are the same. Hence, the solution is also optimal for Problem $\mathcal{E}$ (cf. Corollary 1 in the next section). In such a situation, we call Problem $\mathcal{R}$ as an *exact relaxation* of Problem $\mathcal{E}$.

Nevertheless, ensuring exactness of the relaxation is nontrivial. This means that an optimal solution to Problem $\mathcal{R}$ may be located in the interior of the set in Fig. 2, which is infeasible for Problem $\mathcal{E}$. How to avoid this dilemma will be the topic of the next section.

IV. Exact relaxation: designing an objective function

This section aims to propose an effective method to ensure that Problem $\mathcal{R}$ is an exact relaxation of Problem $\mathcal{E}$, or equivalently that the relaxed constraint in Problem $\mathcal{R}$ is active.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TAES.2021.3131140, IEEE Transactions on Aerospace and Electronic Systems

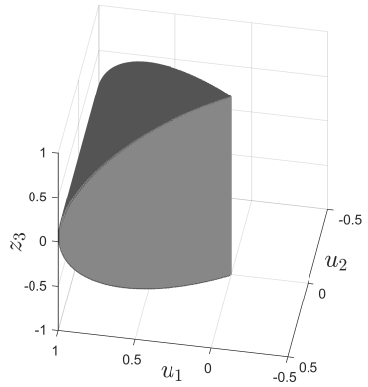

Fig. 2: The feasible set defined by the constraints $u_2 \geq -w_{\max} u_1$, $u_2 \leq w_{\max} u_1$, and $(u_1)^2 + (z_3)^2 \leq 1$ in Problem $\mathcal{R}$.

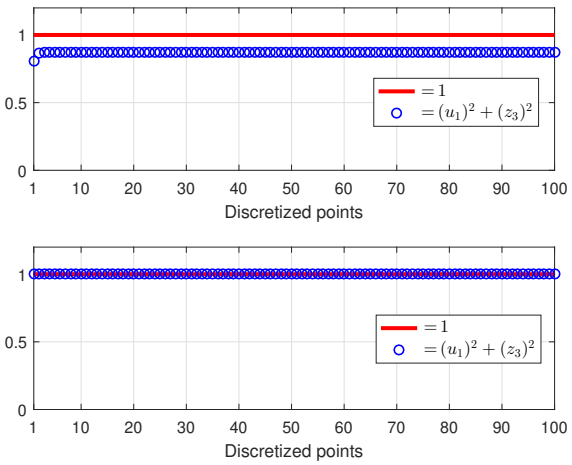

Fig. 3: The effect of the objective function on the status (active or not) of the relaxed constraint $(u_1)^2 + (z_3)^2 \leq 1$. Top plot: $J = \int_{t_0}^{t_f} u_1 \mathrm{d}t$; Bottom plot: $J = \int_{t_0}^{t_f} y \mathrm{d}t$.

We have observed through numerical implementation (numerically solving Problem $\mathcal{R}$ as a nonlinear programming problem) that the relaxed constraint can indeed be inactive. For instance, for a specific example with the objective function set as $J = \int_{t_0}^{t_f} u_1 \mathrm{d}t$, the relaxed constraint is always found inactive, as shown in the top plot of Fig. 3. Interestingly, if we set $J = \int_{t_0}^{t_f} y \mathrm{d}t$, the relaxed constraint immediately becomes active, which is clearly illustrated in the bottom plot of Fig. 3. This phenomenon gives us a hint that setting an appropriate objective function may fulfill the goal of realizing exact relaxation. Note that the influence of the form of an objective function on the status of a relaxed constraint has also been reported in some existing works [40], [2], though there are still not clear guidelines on how to choose the objective function.

In this paper, we propose to select an objective function in the following form for Problem $\mathcal{R}$

$$J = k_1|x(t_f) - x_f| + k_2|y(t_f) - y_f| + k_3 \int_{t_0}^{t_f} f(y)\mathrm{d}t, \tag{24}$$

where $k_1, k_2, k_3 > 0$ are weighting coefficients, (x_f, y_f) is the target position, and $f(y)$ is a convex function of y and is differentiable. Note that in Problem $\mathcal{R}$ the terminal constraints on position are now replaced by penalizing the terminal position error in the above designed objective function.

In the following, we first make the following two mild assumptions

Assumption 3: If any of the two pure control constraints in Problem $\mathcal{R}$ is active in a finite interval, then $\frac{\mathrm{d}f(y)}{\mathrm{d}y} \neq 0$ is satisfied almost everywhere (a.e.) in that interval.

Assumption 4: For Problem $\mathcal{R}$, if $\frac{\mathrm{d}f(y)}{\mathrm{d}y} = 0$ is satisfied in a finite interval, then

(i) all the state constraints are inactive in any finite interval in $[t_0, t_f]$, and
(ii) the target is unreachable.

Remark 2: Note that Assumptions 3-4 are mild. For instance, we may select an $f(y)$ such that the value of $\frac{\mathrm{d}f(y)}{\mathrm{d}y}$ is always nonzero. In this situation, Assumption 3 is always satisfied, and the *if condition* in Assumption 4 never occurs, which implies that Assumption 4 is always satisfied as well. Note that setting $f(y) = y$, which means $\frac{\mathrm{d}f(y)}{\mathrm{d}y} = 1$, is such a selection and is used in the numerical example in Fig. 3. In this paper, we will finally set $f(y) = (y - y_c)^2$ [cf. Eq. (25)], and more explanation on the mildness of the assumptions can be found later in Remarks 3-4.

Then, we can theoretically prove that the designed objective function can make the relaxed constraint active, which is stated in the following theorem

Theorem 1: If the objective function is designed as the one in Eq. (24) and the optimal solution of Problem $\mathcal{R}$ is denoted as $\{z^*; u^*\}$, then under Assumptions 2-4 the condition $(u_1^*)^2 + (z_3^*)^2 = 1$ holds a.e. on $[t_0, t_f]$.

Proof: See the Appendix A.

Theorem 1 establishes that Problem $\mathcal{R}$ is an exact relaxation of Problem $\mathcal{E}$. Based on Theorem 1 and the previously obtained Lemma 1, we get the relationship between Problem $\mathcal{R}$ and Problem $\mathcal{O}$ as follows

Corollary 1: Under Assumptions 1-4, the optimal solution of Problem $\mathcal{R}$ is also optimal to Problem $\mathcal{O}$, and vice versa.

Proof: Denote the optimal cost of Problem $\mathcal{O}$, Problem $\mathcal{E}$, and Problem $\mathcal{R}$ as $J_{\mathcal{O}}^*$, $J_{\mathcal{E}}^*$, and $J_{\mathcal{R}}^*$, respectively (note that their objective functions are the same). Lemma 1 states that Problem $\mathcal{O}$ is equivalent to Problem $\mathcal{E}$ and thus $J_{\mathcal{O}}^* = J_{\mathcal{E}}^*$. The only change from Problem $\mathcal{E}$ to Problem $\mathcal{R}$ is that we relax the mixed state and control constraint in Problem $\mathcal{E}$. This implies that the feasible set of Problem $\mathcal{E}$ is a subset of Problem $\mathcal{R}$, and consequently we have $J_{\mathcal{E}}^* \geq J_{\mathcal{R}}^*$. In addition, under Theorem 1 an optimal solution of Problem $\mathcal{R}$ is always feasible to Problem $\mathcal{E}$, which implies that $J_{\mathcal{R}}^* \geq J_{\mathcal{E}}^*$. Hence, $J_{\mathcal{R}}^* = J_{\mathcal{E}}^*$. Then, we have $J_{\mathcal{R}}^* = J_{\mathcal{O}}^*$. That is to say, the optimal solution of Problem $\mathcal{R}$ is also optimal to Problem $\mathcal{O}$, and vice versa. ■

It is worth noting that if the nonconvex state constraints are not present, Problem $\mathcal{R}$ is convex. Then, Corollary 1 tells us that we only need to solve a single convex Problem $\mathcal{R}$ to get a solution of the original nonconvex Problem $\mathcal{O}$. This is clearly a unique advantage awarded by preserving the nonlinearity inherent in the original nonlinear dynamics. It should be emphasized that the careful design of the objective function in this section is critical to establish Corollary 1.

Due to the existence of the nonconvex state constraints, Problem $\mathcal{R}$ is still nonconvex. To facilitate designing an iterative algorithm to solve the nonconvex Problem $\mathcal{R}$ with guaranteed convergence, we will set $f(y)$ to have the following quadratic form

$$f(y) = (y - y_{\rm c})^2, \tag{25}$$

where $y_{\rm c}$ is a given y profile and is specified according to different mission requirements. This function in the running cost will be used in the rest of this paper. We will show in the numerical examples in Sec. VI that Assumptions 3-4 are easy to hold for the $f(y)$ given in Eq. (25).

Remark 3: Based on the $f(y)$ in Eq. (25), we will explain why Assumptions 3-4 are mild. The objective function can be used to track a reference trajectory y_c, such as lane tracking or lane change in autonomous driving. If $\frac{{\rm d}f(y)}{{\rm d}y} = 2(y - y_c) = 0$ is satisfied or the vehicle completely tracks the reference trajectory y_c in a finite interval, the corresponding control is generally not $w_{\max}$. This implies that Assumption 3 is satisfied. In addition, if our goal is to track a reference trajectory y_c and $\frac{{\rm d}f(y)}{{\rm d}y} = 0$ holds in a finite interval, then all the state constraints will generally be inactive, because having active state constraints usually contradicts the goal of trajectory tracking. Besides, we can make the target unreachable by simply choosing an appropriate t_f. Hence, Assumption 4 is easy to satisfy.

Remark 4: If trajectory tracking is not required, we can set y_c as a certain value (sufficiently large or small, for instance) such that the value of $\frac{{\rm d}f(y)}{{\rm d}y} = 2(y - y_c)$ is always nonzero and consequently Assumptions 3-4 are all always satisfied (cf. Remark 2). This will be seen in the UAV trajectory planning problem in Sec. VI.

V. An Iterative Algorithm with Guaranteed Convergence

In the previous two sections, we have successfully covexified the nonlinear dynamics without posing any approximation, and we have designed an appropriate objective function to achieve exact relaxation (cf. Theorem 1). As a result, we can obtain a solution of Problem $\mathcal{O}$ by solving the relaxed Problem $\mathcal{R}$ (cf. Corollary 1). In this section, we will show how to further convexify the nonconvex state constraints in Problem $\mathcal{R}$ and develop an iterative algorithm which has guaranteed convergence to get a solution of Problem $\mathcal{R}$ (or equivalently Problem $\mathcal{O}$).

A. Discretization

In this subsection, we convert Problem $\mathcal{R}$, which is an infinite-dimensional optimal control problem, into a finite-dimensional nonlinear programming problem by discretization. Specifically, we replace the continuous state $z(t)$ and control $u(t)$ with $(N+1)$ discretized pairs $\{z_i; u_i\}_{i=0,\ldots,N}$, where $z_i = [z_{1i}\ \ z_{2i}\ \ z_{3i}]^T := [x_i\ \ y_i\ \ \theta_{si}]^T$ and $u_i = [u_{1i}\ \ u_{2i}]^T := [\theta_{ci}\ \ \theta_{di}]^T$. Then, the objective function is discretized as

$$J = k_1|x_N - x_f| + k_2|y_N - y_f| + k_3 \sum_{i=0}^{N} \Delta t (y_i - y_{ci})^2, \tag{26}$$

where Δt is computed by $\Delta t = (t_f - t_0)/N$. The system dynamics can be discretized by proper discretization methods and all other constraints are enforced at the discretized points. After discretization, Problem $\mathcal{R}$ becomes

$$\begin{aligned} \text{Problem } \mathcal{D}: \ \min_{\chi}\ & c^T\chi \\ \text{s.t.}\ & H\chi \le q \\ & g^{nc}(x_i, y_i) \le 0,\ i = 0, \ldots, N \\ & \Lambda\chi - b \succeq_K 0, \end{aligned} \tag{27}$$

where the optimization variable χ collects $\{z_i; u_i\}_{i=0,\ldots,N}$ and all additional necessary variables, and K is the direct product of second-order cones. Note that the objective function in Problem $\mathcal{R}$ can be equivalently transformed into a linear function plus some additional constraints. All the convex linear constraints are included in $H\chi \le q$, including the discretized linear dynamics, and all the convex quadratic constraints, including the relaxed constraint imposed at each discretized point, are merged into $\Lambda\chi - b \succeq_K 0$, which means $\Lambda\chi - b \in K$.

The only nonconvexity in Problem $\mathcal{D}$ lies in the constraints $g^{nc}(x_i, y_i) \le 0$, which are obtained by discretizing the nonconvex state constraints in Problem $\mathcal{R}$. How to solve the nonconvex Problem $\mathcal{D}$ will be introduced in the following subsection.

B. An iterative algorithm

It is observed that Problem $\mathcal{D}$ belongs to the class of general nonconvex problems investigated in Ref. [30]. Specifically, Problem $\mathcal{D}$ becomes an SOCP problem if the nonconvex inequality constraints are not present, while the nonconvex inequality constraints have a distinct feature that the constraint functions are all *concave*. As proposed in Ref. [30], we linearize the nonconvex constraints and consequently get the following SOCP problem

$$\begin{aligned} &\text{Problem } \mathcal{D}_c^{(k)}: \\ &\min_{\chi}\ c^T\chi \\ &\text{s.t.}\ H\chi \le q \\ &\quad \nabla g_j^{nc}(x_i^{(k)}, y_i^{(k)})^T \begin{bmatrix} x_i \\ y_i \end{bmatrix} + d_j(x_i^{(k)}, y_i^{(k)}) \le 0 \\ &\quad \Lambda\chi - b \ge_K 0, \end{aligned} \tag{28}$$

where $i = 0, \ldots, N$, $j = 1, \ldots, n^{nc}$, $(x_i^{(k)}, y_i^{(k)})$ is the vehicle position at the kth iteration, $\nabla g_j^{nc}(x_i^{(k)}, y_i^{(k)})$ is the gradient of the constraint function $g_j^{nc}(x_i, y_i)$ at $(x_i^{(k)}, y_i^{(k)})$,

and $d_j(x_i^{(k)}, y_i^{(k)}) = g_j^{nc}(x_i^{(k)}, y_i^{(k)}) - \nabla g_j^{nc}(x_i^{(k)}, y_i^{(k)})^T [x_i^{(k)}\ y_i^{(k)}]^T$. It should be pointed out that in Ref. [30] a trust-region constraint is imposed when the nonconvex constraints are linearized, and it is used for the sole purpose of making the feasible set bounded. Nevertheless, unlike that in Ref. [30], we do not impose any trust-region constraint in Problem $\mathcal{D}_c^{(k)}$, because we can still prove the boundedness of the feasible set of Problem $\mathcal{D}_c^{(k)}$ (see Lemma 2 in the next subsection). Though adding a trust region constraint usually has another purpose of making the linearization locally accurate, in this paper we found that the absence of such a constraint does not affect the convergence of Algorithm 1 designed later. Hence, it is unnecessary to include any trust-region constraint into Problem $\mathcal{D}_c^{(k)}$. Note that this has the benefit of reducing the number of second-order cone constraints and thus saving certain computation time.

Next, to get a solution of Problem $\mathcal{D}$, we design an algorithm to iteratively solve Problem $\mathcal{D}_c^{(k)}$ until convergence is achieved. Detailed steps of the algorithm are described as follows

Algorithm 1:

1) Set $k = 0$ and provide an initial state profile $\{x_i^{(0)}, y_i^{(0)}\}$.
2) At the $(k+1)$th iteration, first compute the parameters $\nabla g_j^{nc}(x_i^{(k)}, y_i^{(k)})$ and $d_j(x_i^{(k)}, y_i^{(k)})$, and then solve Problem $\mathcal{D}_c^{(k)}$ to get the optimal solution $\chi^{(k+1)}$.
3) Check the following stopping criterion:

$$\max_i \left| \begin{bmatrix} x_i^{(k+1)} - x_i^{(k)} \\ y_i^{(k+1)} - y_i^{(k)} \end{bmatrix} \right| \leq \begin{bmatrix} \epsilon_x \\ \epsilon_y \end{bmatrix}, \tag{29}$$

where $\epsilon = [\epsilon_x\ \epsilon_y]^T > 0$ is a user-defined sufficiently small constant vector. If Eq. (29) is not satisfied, update k with $(k+1)$, and then go back to the previous step. Otherwise, go to the next step.
4) The converged optimal solution is $\chi^{(k+1)}$.

In Algorithm 1, Problem $\mathcal{D}_c^{(k)}$ in each iteration can be solved by any SOCP solver in polynomial time. Nevertheless, whether the algorithm can converge within a limited number of iterations is very important if real-time performance is to be attained. We will analyze the convergence of Algorithm 1 in the next subsection.

C. Convergence analysis

This subsection shows that Algorithm 1 is guaranteed to converge, and the converged solution is at least a local optimal solution of Problem $\mathcal{O}$.

First, we make an assumption on strict feasibility of Problem $\mathcal{D}_c^{(k)}$ [30]

Assumption 5: For Problem $\mathcal{D}_c^{(k)}$, there exists a point such that the quadratic constraints are strictly feasible and the linear constraints are feasible.

The above assumption means that Problem $\mathcal{D}_c^{(k)}$ meets the Slater's condition [12], which generally holds as long as the problem is feasible. Next, we get the following two lemmas with proofs given in the appendix

Lemma 2: The feasible set of Problem $\mathcal{D}_c^{(k)}$ is bounded.

Proof: See Appendix B. ■

Lemma 3: The optimal solution of Problem $\mathcal{D}_c^{(k)}$, if exists, is unique.

Proof: See Appendix C. ■

Then, we can also get the following four lemmas for which proofs can be found in Ref. [30] (note that Lemma 2 is needed to prove Lemma 5, while Lemma 3 is needed to prove Lemma 7)

Lemma 4: Any feasible solution to Problem $\mathcal{D}_c^{(k)}$ is also feasible to Problem $\mathcal{D}$.

Proof: See the proof for Lemma 1 in Ref. [30]. ■

Lemma 5: Under Assumption 5, if the optimal solution of Problem $\mathcal{D}_c^{(k)}$ is $\chi^{(k)}$ itself, then $\chi^{(k)}$ is a Karush-Kuhn-Tucker (KKT) solution of Problem $\mathcal{D}$.

Proof: See the proof for Lemma 2 in Ref. [30]. ■

Lemma 6: (*Recursive Feasibility*) If Problem $\mathcal{D}_c^{(k-1)}$ has an optimal solution denoted as $x^{(k)}$, then Problem $\mathcal{D}_c^{(k)}$ has an optimal solution.

Proof: See the proof for Lemma 3 in Ref. [30]. ■

Lemma 7: (*Monotone Decreasing Property*) The solution sequence $\{\chi^{(k)}\}_{k=1,2,\ldots}$ generated by Algorithm 1 satisfies the monotone decreasing property $c^T\chi^{(k+1)} < c^T\chi^{(k)}$.

Proof: See the proof for Lemma 4 in Ref. [30]. ■

Finally, another main result in this paper is given in the following theorem

Theorem 2: Under Assumption 5, for any sufficiently small ϵ in the stopping criterion Algorithm 1 generates a converged solution. Furthermore, under Assumptions 1-4, the converged solution is at least a local optimal solution of Problem $\mathcal{O}$.

Proof: Based on the proof for the theorem in Ref. [30], Lemmas 4-7 can be used to prove that the solution sequence $\{\chi^{(k)}\}$ either terminates at finite steps with $\chi^{(k+1)} = \chi^{(k)}$ or is infinite with an accumulation point χ^* such that the solution of Problem $\mathcal{D}_c^{(*)}$ is χ^*. In both cases, the difference between the final two successive solutions is zero. This implies that the stopping criterion in Algorithm 1 must be satisfied for any ϵ. Hence, Algorithm 1 can generate a converged solution. By applying the *Corollary* in Ref. [30], we can get that the converged solution is at least a local optimal solution of Problem $\mathcal{D}$, i.e., it satisfies the KKT conditions of Problem $\mathcal{D}$.

Furthermore, since Problem $\mathcal{D}$ is just discretized from Problem $\mathcal{R}$, the converged solution is also at least a local optimal solution of Problem $\mathcal{R}$ to the extent of accuracy of the discretization. Under Assumptions 1-4, we can apply Corollary 1 in this paper to get that the optimal solution of Problem $\mathcal{R}$ is also optimal to Problem $\mathcal{O}$. Hence, the converged solution obtained by Algorithm 1 is at least a local optimal solution of Problem $\mathcal{O}$. ■

It can be seen from the above analysis that most of the proofs can be found in Ref. [30], which indicates the significance of the work in the reference. It should be pointed out that in Ref. [30] the convex problem in each iteration, which is Problem $\mathcal{D}_c^{(k)}$ in this paper, is assumed to have a unique solution, and this assumption needs to be used to prove the *Monotone Decreasing Property* (cf. Lemma 7). Nevertheless, we do not need the assumption in this paper since we can prove it, as stated in Lemma 3. Note that this is largely attributed

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TAES.2021.3131140, IEEE Transactions on Aerospace and Electronic Systems

to our design of the *quadratic* objective function [cf. Eq. (25) and the proof for Lemma 3].

Convergence of Algorithm 1 has now been theoretically proved without the need of any trust-region constraint. We will demonstrate in the next section that Algorithm 1 converges very quickly. It should be highlighted that if exact relaxation can not be realized (note that it can occur as shown in Fig. 3), we will not be able to take advantage of the established conclusions in Ref. [30]. On the other hand, if the original nonlinear dynamics are directly linearized at the very beginning, it is generally very challenging to design a convergence-guaranteed algorithm to solve the original nonconvex Problem $\mathcal{O}$. Hence, the proposed methods of convexifying the original nonlinear dynamics with nonlinearity preserved (cf. Sec. III) and realizing exact relaxation by designing an appropriate objective function (cf. Sec. IV) are crucial to achieving the guaranteed convergence of the iterative algorithm.

Remark 5: We note that Assumptions 1-5 are required in Theorem 2. Nevertheless, it should be emphasized that we only need Assumption 5, which is a very mild assumption, to prove the convergence of Algorithm 1, while Assumptions 1-4 are required to prove that the converged solution is a solution of Problem $\mathcal{O}$. In the numerical examples in the next section, we will see that Assumptions 1-4 are easy to be satisfy.

Remark 6: To initiate Algorithm 1, we only need to select an initial state profile $\{x_i^{(0)}, y_i^{(0)}\}$ such that Problem $\mathcal{D}_c^{(0)}$ in the first iteration is feasible or it has at least one solution (the initial state profile itself needs not to be feasible). Note that as long as the first problem, that is Problem $\mathcal{D}_c^{(0)}$, is feasible, all the subsequent problems will be feasible (cf. Lemma 6), and consequently the solution sequence $\{\chi^{(k)}\}$ is guaranteed to converge based on Theorem 2. In addition, we will see in the next section that the algorithm can quickly converge even though a rough and infeasible initial state profile is selected.

VI. Numerical Examples

In this section, we will provide two numerical examples on UAVs and autonomous cars to demonstrate the high performance of Algorithm 1. A desktop with Intel Core i7-6700 3.40 GHz and the software ECOS [17] for solving SOCP problems are used to obtain the numerical results. The computation times of running the algorithm in a single-board computer named Raspberry Pi 3 Model B+ (which has a 1.4GHz quad-core processor) will also be reported.

A. Trajectory planning of UAVs with avoidance zones

We consider a UAV with an initial state $(x_0, y_0, \theta_0) = (0, 0, 0)$ and a constant velocity of $v = 1$ m/s. It is required to reach a target at $(x_f, y_f) = (25, 6)$ m, and at the same time keep away from three avoidance zones in either circular or elliptic shapes (cf. Fig. 4). The maximum angular velocity $w_{\max}$ is set to be 8 deg/s. In addition, the UAV is required to finally enter a narrow cone and approach the target from a specified direction (see the dotted blue lines in Fig. 4). The approach cone has a half angle of 3 deg and the angle between the cone's centerline and the horizon line is 20 deg. We can enforce two linear state constraints, which are related to the boundaries of the cone, to meet the requirement. A total number of 101 ($N = 100$) discretized points are used. In Algorithm 1, we set the initial state profile $\{x_i^{(0)}, y_i^{(0)}\}$ as three line segments as shown by the dash-dotted line in Fig. 4 and set $y_c = -1$ m, $k_1 = k_2 = 100$, $k_3 = 1$, $\epsilon = [0.1 \;\; 0.1]^T$ m. The time of flight is set as $t_f = 26.5$ s. It is worth pointing out that in this example our goal is to find a feasible trajectory to reach the target. Hence, we can first let $k_3 = 1$ and then select sufficiently large k_1 and k_2 to penalize the terminal position error. This means that setting k_1 and k_2 as other large values, such as 1000 or 2000, will also work and the solution is not sensitive to their values. In addition, y_c can be set as other constants or time-varying profiles as well as long as the value of $(y - y_c)$ is always nonzero (cf. Remark 4 for the reason).

The solution of Problem $\mathcal{O}$ obtained by Algorithm 1 is plotted in Figs. 4-6. It can be seen that all the state constraints and control constraints are satisfied and the relaxed constraint is always active. To demonstrate the performance of Algorithm 1 in terms of convergence and efficiency, we give the difference of (x, y) between consecutive iterations and the computation time in each iteration in Table I. It shows that Algorithm 1 converges in two iterations for $\epsilon = [0.1 \;\; 0.1]^T$ and the total computation time is 85.4 ms (it is 971.7 ms when the algorithm is run in the single-board computer). We also make the algorithm run two more iterations after convergence is achieved (see $k = 3, 4$ in the table). It is to just show that Algorithm 1 can still achieve convergence even if we set a much smaller ϵ in the stopping criterion. Note that this is expected since Algorithm 1 is proved to converge for any sufficiently small ϵ (cf. Theorem 2). In addition, Table I shows that the rate of convergence of Algorithm 1 is very high, because the difference is always decreased by roughly two orders of magnitude in each iteration. This example has shown the quick convergence and high efficiency of the algorithm.

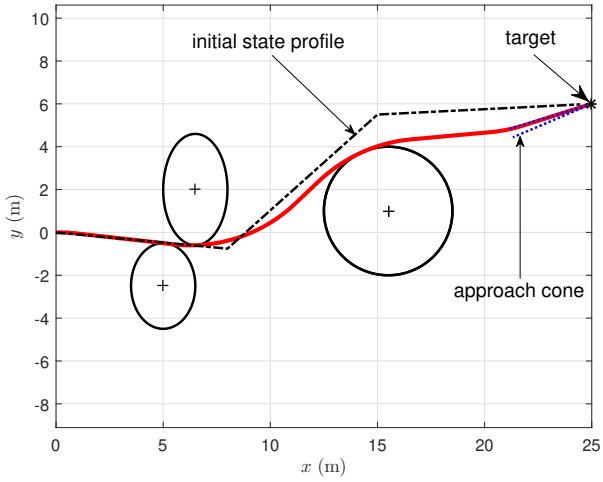

Fig. 4: The flight trajectory of a UAV with multiple avoidance zones.

It is worth noting that the Assumptions 1-4 required in Theorem 2 are all satisfied in this example. Recall that Assumptions 1-2 are made on Problem $\mathcal{O}$, while Assumptions 3-4

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TAES.2021.3131140, IEEE Transactions on Aerospace and Electronic Systems

are put on Problem $\mathcal{R}$. Since Corollary 1 tells that the optimal solutions of Problem $\mathcal{O}$ and Problem $\mathcal{R}$ are the same, we will verify Assumptions 1-4 based on the solution of Problem $\mathcal{O}$ shown in Figs. 4-5. Specifically, the top plot in Fig. 5 shows that the magnitude of the heading angle is always less than 90 deg, which verifies Assumption 1. We can see in Figs. 4-5 that the collision avoidance constraints are inactive whenever the control constraints are active, and the control constraints are inactive when the approach cone constraints are active in the last 3 seconds approximately. This phenomenon is consistent with Assumption 2. To check Assumptions 3-4, we see from Fig. 4 that $\frac{\mathrm{d}f(y)}{\mathrm{d}y} = 2(y-y_c) = 2(y+1)$ can not be zero since y is always not equal to -1. This implies that Assumptions 3-4 are automatically satisfied.

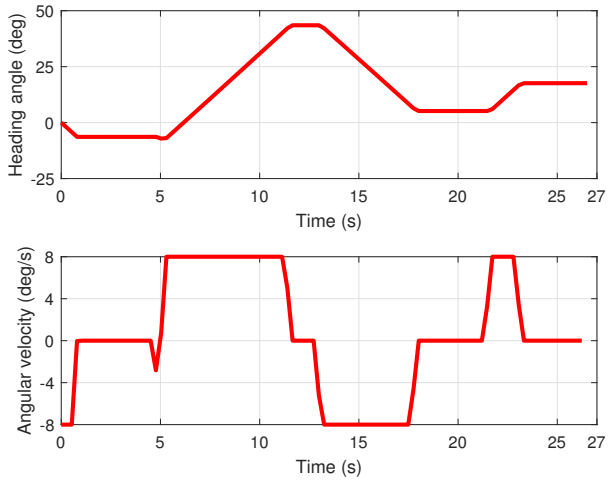

Fig. 5: The heading angle and angular velocity histories of a UAV with multiple avoidance zones.

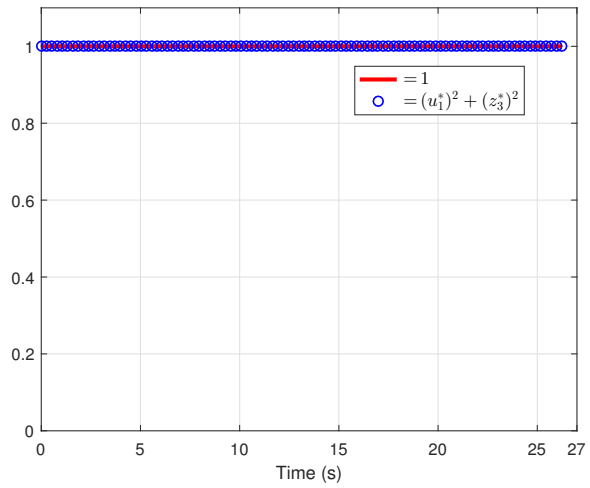

Fig. 6: The status of the relaxed constraint for a UAV with multiple avoidance zones.

TABLE I: The difference of (x, y) between consecutive iterations in Algorithm 1 and the computation time for solving Problem $\mathcal{D}_c^{(k)}$ in each iteration for the UAV trajectory planning problem. Additional two iterations ($k = 3, 4$) are given.

k	$\max_i \lvert\Delta x_i\rvert$ (m)	$\max_i \lvert\Delta y_i\rvert$ (m)	computation time (ms)
1	0.763867	1.274738	23.3
2	0.009435	0.018163	62.1
3	0.000133	0.000174	55.8
4	0.000001	0.000001	57.9

B. Lane change of autonomous cars

In this subsection, an autonomous car with a constant velocity of $v = 7$ m/s and $(x_0, y_0, \theta_0) = (0, 1.75, 0)$ m is required to make a lane change and move towards the target at $(x_f, y_f) = (100, 5.25)$ m. A reference trajectory is set as

$$y_c = \begin{cases} 1.75, & x \in [0, 70) \\ 5.25, & x \in [70, 100], \end{cases}$$

which means that the vehicle is required to make a lane change around $x = 70$ m (see Fig. 7). The maximum angular velocity $w_{\max}$ is set as 15 deg/s and the car needs to avoid an obstacle along the way. The initial state profile $\{x_i^{(0)}, y_i^{(0)}\}$ is selected as a straight line connecting (x_0, y_0) and (x_f, y_f). In addition, we set $k_1 = k_2 = 1$, $k_3 = 1000$, $N = 100$, $\epsilon = [0.1\ 0.1]^T$ m, and $t_f = 14$ s. Note that our main goal is to make the car track the reference trajectory. Hence, we can first set $k_1 = k_2 = 1$ and then select a sufficiently large k_3 to penalize the tracking error. Though other values of k_3 can be selected, it may slightly affect the solution in terms of the tracking error. The solution obtained by Algorithm 1 is shown in the red solid lines in Figs. 7-8, indicating that the car successfully avoids the obstacle and completes the lane change without violating the control constraints.

In this example, we will also show that Assumptions 1-4 are satisfied based on the solution shown in Figs. 7-8. It is obvious that the Assumptions 1-2 are satisfied since the heading angle is always less than 90 deg and using the maximum angular velocity can not make the car ride on the boundary of the obstacle. Next, we will verify Assumptions 3-4. Note that when the car completely tracks the center lines of lanes we have $\frac{\mathrm{d}f(y)}{\mathrm{d}y} = 2(y - y_c) = 0$. Figure 8 shows that the control constraints are active in some finite intervals. But, in these intervals the car keeps changing its heading angle and can not track the center line of lanes, or in other words, $\frac{\mathrm{d}f(y)}{\mathrm{d}y} \neq 0$ holds. Thus, Assumption 3 is satisfied. For Assumption 4, the (i) part is satisfied since the top plot of Fig. 7 shows that the obstacle avoidance constraint is not active when $\frac{\mathrm{d}f(y)}{\mathrm{d}y} = 0$. To make the (ii) part hold, we can choose an appropriate t_f so that the target is unreachable. Since $\sqrt{(x_f - x_0)^2 + (y_f - y_0)^2}/v = 14.29$ s, any t_f smaller than 14.29 s can make the target unreachable (note that we set $t_f = 14$ s). The above discussion indicates that the Assumptions 1-4 in Theorem 2 are all satisfied. Hence, based

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TAES.2021.3131140, IEEE Transactions on Aerospace and Electronic Systems

on Theorem 2 it is guaranteed that Algorithm 1 can converge to an optimal solution of Problem $\mathcal{O}$.

Table II shows that Algorithm 1 converges quickly in three iterations and the computation cost is about 118.0 ms (it is 1035.6 ms when the algorithm is run in the single-board computer). Two more iterations (see $k = 4, 5$ in the table) are also given to show the capacity of Algorithm 1 to converge for a much smaller ϵ. This again shows the quick convergence and high efficiency of the algorithm. It is worth pointing out that when the obstacle is not present, Algorithm 1 only needs to solve one single SOCP problem to get a solution of the nonconvex Problem $\mathcal{O}$, and the computation time is just 14.5 ms. The solution is shown by the blue dashed lines in Figs. 7-8.

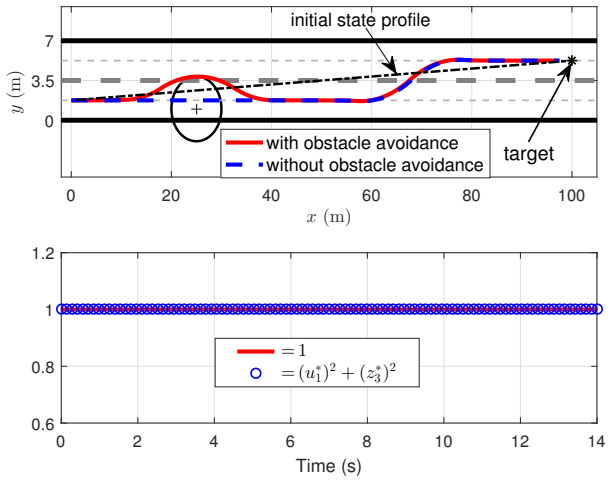

Fig. 7: Top plot: The lane-change trajectories of a car with and without obstacle avoidance constraint. Bottom plot: The status of the relaxed constraint when the obstacle avoidance constraint is either present or not present.

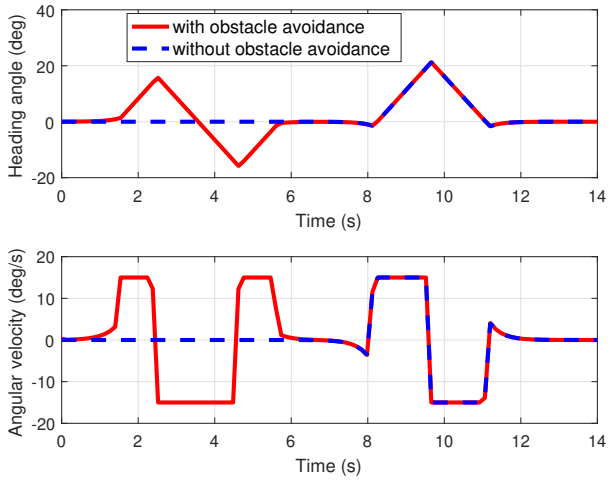

Fig. 8: The lane-change heading angle and angular velocity histories of a car with and without obstacle avoidance constraint.

TABLE II: The difference of (x, y) between consecutive iterations in Algorithm 1 and the computation time for solving Problem $\mathcal{D}_c^{(k)}$ in each iteration for the lane change problem. Additional two iterations ($k = 4, 5$) are given.

k	$\max_i \lvert\Delta x_i\rvert$ (m)	$\max_i \lvert\Delta y_i\rvert$ (m)	computation time (ms)
1	2.92051878	2.17858957	14.8
2	0.07984054	0.37925766	57.6
3	0.00056288	0.00407360	45.6
4	0.00000605	0.00004290	48.5
5	0.00000007	0.00000046	47.6

VII. Conclusions

This paper investigates the trajectory planning problem for a class of nonlinear system which can represent the motion of autonomous cars, mobile robots, UAVs, missiles, etc. Convex state/control constraints and nonconvex state constraints with concave constraint functions are also incorporated into the corresponding OCP formulation. The main contribution of this paper lies in the proposed approach of convexifying the nonlinear dynamics without any approximation, together with selecting an appropriate objective function to ensure exact relaxation under mild assumptions. Finally, we can design an iterative algorithm with guaranteed convergence to an optimal solution of the original problem (which has nonconvex dynamics and nonconvex state constraints). The proposed approach is novel since the designed algorithm is proved to converge only under an assumption of the Slater's condition and the proof is completed without using any trust-region constraint. Quick convergence and high efficiency of the algorithm are clearly demonstrated by implementing the algorithm on both a desktop and a single-board computer (Raspberry Pi 3 Model B+) for trajectory planning of a UAV with multiple avoidance zones and lane change of an autonomous car with an obstacle avoidance requirement.

Appendix A: Proof of Theorem 1

First, we make an equivalent transformation of the objective function in Problem $\mathcal{R}$ so that the resulting problem is in standard form. Specifically, by introducing two state variables (ξ_x, ξ_y) and two control variables (η_x, η_y), we have a new objective function

$$J = k_1\xi_x(t_f) + k_2\xi_y(t_f) + k_3\int_{t_0}^{t_f} f(y)\mathrm{d}t, \tag{A1}$$

where ξ_x and ξ_y are governed by

$$\dot{\xi_x}(t) = \eta_x, \quad \dot{\xi_y}(t) = \eta_y \tag{A2}$$

and two additional terminal constraints are

$$|x(t_f) - x_f| \le \xi_x(t_f), \quad |y(t_f) - y_f| \le \xi_y(t_f). \tag{A3}$$

The above idea of equivalent transformation was previously used in Ref. [40]. It was also shown in the reference that the costate variables associated with the state variables ξ_x and

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TAES.2021.3131140, IEEE Transactions on Aerospace and Electronic Systems

ξ_y, denoted as p_{ξ_x} and p_{ξ_y} in this paper, are zero for all t. This condition will be used to simplify the notation when we give the necessary optimality conditions for Problem $\mathcal{R}$. Based on the optimal control theory [41], the Hamiltonian and Lagrangian functions are defined as follows

$$H = p_0 k_3 f(y) + p_x v u_1 + p_y v z_3 + p_{\theta_s} u_2 \tag{A4}$$

$$L = H + \lambda_u(1 - u_1^2 - z_3^2) + \lambda_w^-(u_2 + w_{\max} u_1) + \lambda_w^+(w_{\max} u_1 - u_2) + \lambda_g^T(-g), \tag{A5}$$

where $p_0 \leq 0$ is a constant, $p = [p_x \ p_y \ p_{\theta_s}]^T \in \mathbb{R}^3$ is the costate vector, g contains g^c and g^{nc}, and λ_u, λ_{u_1}, λ_w^-, λ_w^+, and $\lambda_g \in \mathbb{R}^{(n^c+n^{nc})}$ are the Lagrangian multipliers. The Lagrangian is formed by directly adjoining all the constraints into the Hamiltonian with the help of Lagrangian multipliers. This idea of defining the Lagrangian brings convenience to describe the necessary optimality conditions that an optimal solution should satisfy, and it is called the *direct adjoining approach* in Ref. [41]. Note that $p_{\xi_x} = p_{\xi_y} = 0$ is used when the Hamiltonian is defined.

Then, the necessary optimality conditions include [41]:

1) the nontriviality condition:

$$[p_0 \ p^T \ \lambda_g^T]^T \neq 0, \ \forall t \in [t_0, t_f]. \tag{A6}$$

2) the costate equations:

$$\dot{p}_x = -\partial_x L = \lambda_g^T \frac{\partial g}{\partial x} \tag{A7}$$

$$\dot{p}_y = -\partial_y L = -p_0 k_3 \frac{\mathrm{d}f}{\mathrm{d}y} + \lambda_g^T \frac{\partial g}{\partial y} \tag{A8}$$

$$\dot{p}_{\theta_s} = -\partial_{\theta_s} L = -p_y v + 2\lambda_u z_3. \tag{A9}$$

3) the stationary conditions:

$$\partial_{u_1} L = p_x v - 2\lambda_u u_1 + \lambda_w^- w_{\max} + \lambda_w^+ w_{\max} = 0 \tag{A10}$$

$$\partial_{u_2} L = p_{\theta_s} + \lambda_w^- - \lambda_w^+ = 0. \tag{A11}$$

4) the complementary slackness conditions:

$$\lambda_u \geq 0, \ \lambda_u(1 - u_1^2 - z_3^2) = 0 \tag{A12}$$

$$\lambda_w^- \geq 0, \ \lambda_w^-(u_2 + w_{\max} u_1) = 0 \tag{A13}$$

$$\lambda_w^+ \geq 0, \ \lambda_w^+(w_{\max} u_1 - u_2) = 0 \tag{A14}$$

$$\lambda_g \geq 0, \ \lambda_g^T g = 0. \tag{A15}$$

4) the transversality conditions:

$$p_{\xi_x}(t_f) = k_1 p_0 + \mu_x^- + \mu_x^+ = 0 \tag{A16}$$

$$p_{\xi_y}(t_f) = k_2 p_0 + \mu_y^- + \mu_y^+ = 0 \tag{A17}$$

$$p_x(t_f) = \mu_x^- - \mu_x^+ \tag{A18}$$

$$p_y(t_f) = \mu_y^- - \mu_y^+. \tag{A19}$$

where $\mu_x^-, \mu_x^+, \mu_y^-, \mu_y^+ \geq 0$ are multipliers associated with the terminal constraints in Eq. (A3).

6) the invariant Hamiltonian condition:

$$H = c, \tag{A20}$$

where c is a constant. Note that Eq. (A20) is obtained since the Hamiltonian is not explicitly dependent on t.

We will prove the theorem by contradiction. Assume that there exists a finite interval $[t_m, t_n] \subset [t_0, t_f]$ over which $[u_1^*]^2 + [z_3^*]^2 < 1$ is satisfied. Then, Eq. (A12) yields $\lambda_u = 0$ in that interval. In the following, two cases will be considered based on the activation of the angular velocity constraints.

Case 1: The angular velocity constraints are always active over $[t_m, t_n]$.

In this case, without loss of generality we suppose that $u_2 \leq w_{\max} u_1$ is active over $[t_m, t_n]$, that is

$$w_{\max} u_1 - u_2 = 0, \tag{A21}$$

or the vehicle moves at the maximum positive angular velocity. Then, the other constraint is not active and thus Eq. (A13) results in $\lambda_w^- = 0$. In addition, under Assumption 2, all the state constraints are inactive, i.e., $g < 0$. This, together with Eq. (A15), gives $\lambda_g = 0$.

Substituting $\lambda_u = \lambda_w^- = \lambda_g = 0$ into Eqs. (A7)-(A11) yields

$$\dot{p}_x = 0 \tag{A22}$$

$$\dot{p}_y = -p_0 k_3 \frac{\mathrm{d}f}{\mathrm{d}y} \tag{A23}$$

$$\dot{p}_{\theta_s} = -p_y v \tag{A24}$$

$$p_x v + \lambda_w^+ w_{\max} = 0 \tag{A25}$$

$$p_{\theta_s} - \lambda_w^+ = 0, \tag{A26}$$

where $\mathrm{d}f/\mathrm{d}y \neq 0$ due to Assumption 3. It can be derived from Eqs. (A25)-(A26) that

$$p_{\theta_s} = -\frac{v p_x}{w_{\max}}. \tag{A27}$$

Differentiating both sides of Eq. (A27) and utilizing Eqs. (A22) and (A24) can generate

$$p_y = \frac{\dot{v}}{w_{\max} v} p_x. \tag{A28}$$

Again, differentiating the above equation and utilizing Eq. (A23) results in

$$\frac{\mathrm{d}}{\mathrm{d}t}\left(\frac{\dot{v}}{v}\right) p_x = -w_{\max} p_0 k_3 \frac{\mathrm{d}f}{\mathrm{d}y}. \tag{A29}$$

If v is time-varying in $[t_p, t_q]$, the term $\frac{\mathrm{d}}{\mathrm{d}t}(\frac{\dot{v}}{v})$ is time-varying as well. Since both p_x and p_0 are constants, Eq. (A29) implies that $p_x = p_0 = 0$. Then, Eqs. (A27)-(A28) with $p_x = 0$ yield $p_{\theta_s} = p_y = 0$.

If v is a constant, $\dot{v} = 0$. Substituting it into Eqs. (A28)-(A29) gives $p_y = p_0 = 0$, and then Eq. (A9) implies that p_{θ_s} is a constant. We will prove $p_{\theta_s} = p_x = 0$ in the following. First, we claim that for given (x_0, y_0) and (x_f, y_f) the vehicle does not move at $w_{\max}$ all the way from t_0 to t_f. Without loss of generality, suppose that the angular velocity constraints are active over the interval $[t_m, t_i]$ with $t_i \geq t_n$ and subsequently become inactive over a finite interval $[t_i, t_j]$. Then we can prove that p_{θ_s} is also a constant over $[t_n, t_i]$ (details are omitted to save space) and p_{θ_s} is zero over $[t_i, t_j]$ [cf. Eq. (A11)]. Since there are no jumps on p_{θ_s} (no state constraints exist on z_3), it is straightforward to get $p_{\theta_s} = 0$ in $[t_m, t_n]$. As a result, $p_x = 0$ by applying Eq. (A27).

In this case, we have obtained $p_0 = p_x = p_y = p_{\theta_s} = \lambda_g = 0$ in $[t_m, t_n]$ no matter v is time-varying or a constant. This contradicts the nontriviality condition in Eq. (A6).

Case 2: There exists a finite interval $[t_p, t_q] \subset [t_m, t_n]$ over which the angular velocity constraints are inactive.

Based on Eqs. (A13)-(A14), we can get $\lambda_w^- = \lambda_w^+ = 0$. Then, Eqs. (A10)-(A11) directly result in $p_x = p_{\theta_s} = 0$. When $p_{\theta_s} = 0$, Eq. (A9) gives $p_y = 0$. In the following, we will show $p_0 = \lambda_g = 0$.

Since $p_x = p_y = 0$, Eqs. (A7)-(A8) become

$$\lambda_g^T \frac{\partial g}{\partial x} = 0 \tag{A30}$$

$$-p_0 k_3 \frac{\mathrm{d}f}{\mathrm{d}y} + \lambda_g^T \frac{\partial g}{\partial y} = 0. \tag{A31}$$

We first consider the situation when $\mathrm{d}f/\mathrm{d}y \neq 0$. In this situation, if all the state constraints are inactive, it is straightforward to get $\lambda_g = 0$ from Eq. (A15) and further get $p_0 = 0$ from Eqs. (A30)-(A31). If one of the state constraints is active (note that at most one can be active under Assumption 2), i.e, $g_i \leq 0$ is active, then all the components of λ_g except the ith component are zero. When $\partial g_i/\partial x \neq 0$, Eq. (A30) generates $(\lambda_g)_i = 0$ (thus $\lambda_g = 0$) and consequently Eq. (A31) gives $p_0 = 0$. Note that if $\partial g_i/\partial x = 0$ or equivalently g_i is independent on x, we can slightly rotate the coordinate frame to equivalently rewrite Problem $\mathcal{R}$ so that the new g_i is dependent on x and thus $\partial g_i/\partial x \neq 0$ is satisfied.

Next, we consider the situation when $\mathrm{d}f/\mathrm{d}y = 0$. Under Assumption 4, all the state constraints are inactive ($\lambda_g = 0$) and thus $p_x = p_y = 0$ for all $t \in [t_0, t_f]$. As a result, $p_x(t_f) = p_y(t_f) = 0$. Furthermore, the target is unreachable, which implies that at least one of the four constants $\mu_x^-, \mu_x^+, \mu_y^-, \mu_y^+$ is zero. Without loss of generality, assume $\mu_x^- = 0$. Substituting $p_x(t_f) = \mu_x^- = 0$ into Eq. (A18) yields $\mu_x^+ = 0$. Then, we can get $p_0 = 0$ from Eq. (A16).

Based on the above discussion, we have proved that $p_x = p_y = p_\theta = p_0 = \lambda_g = 0$ holds at some t. This contradicts the nontriviality condition in Eq. (A6).

In sum, with the contradiction in both cases, the original hypothesis does not hold and we have that $[u_1^*]^2 + [z_3^*]^2 = 1$ is satisfied a.e. on $[t_0, t_f]$.

Appendix B: Proof of Lemma 2

We first prove that the feasible set of Problem $\mathcal{R}$ is bounded. Based on the constraints $(u_1)^2 + (z_3)^2 \leq 1$ and $|u_2| \leq w_{\max} u_1$, it is straightforward to get that the controls are bounded by

$$|u_1| \leq 1, \quad |u_2| \leq w_{\max}. \tag{B1}$$

For the linear control system in Problem $\mathcal{R}$, we compute the state solution as

$$z(t) = e^{A(t-t_0)} z_0 + \int_{t_0}^{t_f} e^{A(t-s)} B u(s) \mathrm{d}s. \tag{B2}$$

Then, we have

$$\|z(t)\| \leq \|e^{A(t-t_0)}\| \|z_0\| + \int_{t_0}^{t_f} \|e^{A(t-s)}\| \|B\| \|u(s)\| \mathrm{d}s.$$

Some items in the right side of the above inequality satisfy the following relationships

$$\begin{aligned} \|e^{A(t-t_0)}\| &= \sigma_{\max}(e^{A(t-t_0)}) \\ &\leq \Big(\frac{v^2\bar{t}^2 + v\bar{t}\sqrt{v^2\bar{t}^2+4}}{2}\Big)^{0.5} \end{aligned} \tag{B3}$$

$$\|B\| = \sigma_{\max}(B) = \sqrt{\max(1, v^2)} \tag{B4}$$

$$\|u\| \leq \sqrt{1 + w_{\max}^2}, \tag{B5}$$

where $\sigma_{\max}$ is the maximum singular value, and $\bar{t} = t_f - t_0$. The inequality in Eq. (B3) is obtained using a condition $t - t_0 \leq \bar{t}$ for $t \in [t_0, t_f]$. Then, it is ready to get

$$\begin{aligned} \|z(t)\| \leq \Big(\frac{v^2\bar{t}^2 + v\bar{t}\sqrt{v^2\bar{t}^2+4}}{2}\Big)^{0.5} &[\|z_0\| \\ + \bar{t}\sqrt{\max(1, v^2)(1 + w_{\max}^2)}&], \end{aligned} \tag{B6}$$

which implies that the states are bounded for given t_f, z_0, and v. Those state inequality constraints in Problem $\mathcal{R}$ may narrow the state space, but do not change the boundedness of the states. Hence, the feasible set of Problem $\mathcal{R}$ is bounded. Furthermore, the feasible set of Problem $\mathcal{D}$ is bounded since the discretization process generally does not change the boundedness of the feasible set. From Problem $\mathcal{D}$ to Problem $\mathcal{D}_c^{(k)}$, linearizing the state-related inequality constraints will also not change the boundedness of the feasible set, and thus the proof is completed.

Appendix C: Proof of Lemma 3

The feasible set of Problem $\mathcal{D}_c^{(k)}$ is convex since it is a convex problem. According to Proposition 2.1.2 in Ref. [42], if we can prove that the objective function J in Eq. (26) is strictly convex, Problem $\mathcal{D}_c^{(k)}$ has at most one solution. In other words, for any two feasible pairs $\tilde{\mathrm{x}}$ and $\hat{\mathrm{x}}$, we need to prove

$$\vartheta J(\tilde{\mathrm{x}}) + (1-\vartheta) J(\hat{\mathrm{x}}) > J(\vartheta\tilde{\mathrm{x}} + (1-\vartheta)\hat{\mathrm{x}}) \tag{C1}$$

for $\tilde{\mathrm{x}} \neq \hat{\mathrm{x}}$ and $\vartheta \in (0,1)$. We notice that J is dependent on y. Note that different x certainly implies a different y, because all other states and controls are uniquely determined when y is given, as can be seen from the system dynamics. Denote $\tilde{y}$ and $\hat{y}$ be the elements in $\tilde{\mathrm{x}}$ and $\hat{\mathrm{x}}$, respectively. To prove (C1), we compute

$$\begin{aligned} &\vartheta J(\tilde{\mathrm{x}}) + (1-\vartheta) J(\hat{\mathrm{x}}) - J(\vartheta\tilde{\mathrm{x}} + (1-\vartheta)\hat{\mathrm{x}}) \\ \geq & k_3 \Delta t (\vartheta \sum_{i=0}^{N} (\tilde{y}_i - y_{\mathrm{c}i})^2 + (1-\vartheta) \sum_{i=0}^{N} (\hat{y}_i - y_{\mathrm{c}i})^2 \\ & - \sum_{i=0}^{N} [\vartheta\tilde{y}_i + (1-\vartheta)\hat{y}_i - y_{\mathrm{c}i}]^2) \\ = & k_3 \Delta t\, \vartheta(1-\vartheta) \sum_{i=0}^{N} (\tilde{y}_i - \hat{y}_i)^2 > 0, \end{aligned} \tag{C2}$$

where $\vartheta \in (0,1)$ and $\tilde{y} \neq \hat{y}$ are used in the last inequality. Hence, the objective function J is strictly convex and consequently the solution of Problem $\mathcal{D}_c^{(k)}$, if it exists, is unique.

REFERENCES

[1] L. Rodrigues, "A unified optimal control approach for maximum endurance and maximum range," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 54, no. 1, pp. 385–391, 2018.

[2] X. Liu, Z. Shen, and P. Lu, "Exact convex relaxation for optimal flight of aerodynamically controlled missiles," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 52, no. 4, pp. 1881–1892, 2016.

[3] B. Acikmese, J. Carson, and L. Blackmore, "Lossless convexification of nonconvex control bound and pointing constraints of the soft landing optimal control problem," *IEEE Transactions on Control Systems Technology*, vol. 21, no. 6, pp. 2104–2113, 2013.

[4] H. Yang and H. Baoyin, "Fuel-optimal control for soft landing on an irregular asteroid," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 51, no. 3, pp. 1688–1697, 2015.

[5] J. Nocedal and S. J. Wright, *Numerical Optimization*, 2nd ed. New York: Springer-Verlag, 2006.

[6] J. T. Betts, *Practical Methods for Optimal Control and Estimation Using Nonlinear Programming*, 2nd ed. Philadelphia, PA: SIAM, 2010.

[7] C. Goerzen, Z. Kong, and B. Mettler, "A survey of motion planning algorithms from the perspective of autonomous UAV guidance," *Journal of Intelligent & Robotic Systems*, vol. 57, pp. 65–100, 2010.

[8] F. Allaire, G. Labonte, M. Tarbouchi, and V. Roberge, "Recent advances in unmanned aerial vehicle real-time trajectory planning," *Journal of Unmanned Vehicle Systems*, vol. 7, no. 4, pp. 259–295, 2019.

[9] B. Paden, M. Cap, 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.

[10] L. Cheng, Z. Wang, F. Jiang, and C. Zhou, "Real-time optimal control for spacecraft orbit transfer via multiscale deep neural networks," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 55, no. 5, pp. 2436–2450, 2019.

[11] S. M. Nolan, C. A. Smith, and J. D. Wood, "Real-time onboard trajectory optimization using indirect methods." *AIAA Scitech 2021 Forum*, AIAA 2021-0106, 2021.

[12] S. Boyd and L. Vandenberghe, *Convex Optimization*. New York: Cambridge University Press, 2004.

[13] U. Eren, A. Prach, B. B. Kocer, S. V. Rakovic, E. Kayacan, and B. Acikmese, "Model predictive control in aerospace systems: Current state and opportunities," *Journal of Guidance, Control, and Dynamics*, vol. 40, no. 7, pp. 1541–1566, 2017.

[14] A. Zanelli, A. Domahidi, J. Jerez, and M. Morari, "FORCES NLP: an efficient implementation of interior-point methods for multistage nonlinear nonconvex programs," *International Journal of Control*, vol. 93, no. 1, pp. 13–29, 2017.

[15] Y. Wang and S. Boyd, "Fast model predictive control using online optimization," *IEEE Transactions on Control System Technology*, vol. 18, no. 2, pp. 267–278, 2010.

[16] J. Mattingley and S. Boyd, "CVXGEN: A code generator for embedded convex optimization," *Optimization and Engineering*, vol. 13, no. 1, pp. 1–27, 2012.

[17] A. Domahidi, E. Chu, and S. Boyd, "ECOS: an SOCP solver for embedded system," in *European Control Conference, IEEE, Zurich, Switzerland*, Jul. 2013, pp. 3071–3076.

[18] A. Domahidi, A. U. Zgraggen, M. N. Zeilinger, M. Morari, and C. N. Jones, "Efficient interior point methods for multistage problems arising in receding horizon control," in *51st IEEE Conference on Decision and Control, Hawaii, USA*, 2012.

[19] D. Dueri, B. Acikmese, D. P. Scharf, and M. W. Harris, "Customized real-time interior-point methods for onboard powered-descent guidance," *Journal of Guidance, Control, and Dynamics*, vol. 40, no. 2, pp. 197–212, 2017.

[20] P. Lu, "Introducing computational guidance and control," *Journal of Guidance, Control, and Dynamics*, vol. 40, no. 2, pp. 193–193, 2017.

[21] B. Acikmese and L. Blackmore, "Lossless convexification for a class of optimal problems with nonconvex control constraints," *Automatica*, vol. 47, no. 2, pp. 341–347, 2011.

[22] M. W. Harris and B. Acikmese, "Lossless convexification of nonconvex optimal control problems for state constrained linear systems," *Automatica*, vol. 50, no. 9, pp. 2304–2311, 2014.

[23] A. Ben-Tal and A. Nemirovski, *Lectures on Modern Convex Optimization: Analysis, Algorithms, and Engineering Applications*. Philadelphia, PA: Society for Industrial and Applied Mathematics, 2001, ch. 2-3.

[24] B. Acikmese and S. R. Ploen, "Convex programming approach to powered descent guidance for mars landing," *Journal of Guidance, Control, and Dynamics*, vol. 30, no. 5, pp. 1353–1366, 2007.

[25] L. Blackmore, B. Acikmese, and D. P. Scharf, "Minimum landing error powered descent guidance for mars landing using convex optimization," *Journal of Guidance, Control, and Dynamics*, vol. 33, no. 4, pp. 1161–1171, 2010.

[26] M. W. Harris and B. Acikmese, "Maximum divert for planetary landing using convex optimization," *Journal of Optimization Theory and Applications*, vol. 162, no. 3, pp. 975–995, 2013.

[27] B. Acikmese, J. Casoliva, S. Moha, and A. Johnson, "Flight testing of trajectories computed by G-FOLD: Fuel optimal large divert guidance algorithm for planetary landing." The 23rd AAS/AIAA Space Flight Mechanics Meeting, 2013.

[28] D. P. Scharf, B. Acikmese, D. Dueri, J. Benito, and J. Casoliva, "Implementation and experimental demonstration of onboard powered-descent guidance," *Journal of Guidance, Control, and Dynamics*, vol. 40, no. 2, pp. 213–229, 2017.

[29] J. M. Carson *et al.*, "The splice project: Continuing NASA development of GN&C technologies for safe and precise landing." *AIAA Scitech 2019 Forum*, AIAA 2019-0660, 2019.

[30] X. Liu and P. Lu, "Solving nonconvex optimal control problems by convex optimization," *Journal of Guidance, Control, and Dynamics*, vol. 37, no. 3, pp. 750–765, 2014.

[31] Y. Mao, M. Szmuk, and B. Acikmese, "Successive convexification of non-convex optimal control problems and its convergence properties." *IEEE 55th Conference on Decision and Control (CDC)*, IEEE, Las Vegas, VN, Dec. 2016, pp. 3636–3641.

[32] X. Liu, P. Lu, and B. Pan, "Survey of convex optimization for aerospace applications," *Astrodynamics*, vol. 1, no. 1, pp. 23–40, 2017.

[33] L. Consolini and C. M. Verrelli, "Learning control in spatial coordinates for the path-following of autonomous vehicles," *Automatica*, vol. 50, pp. 1867–1874, 2014.

[34] H. Teimoori and A. V. Savkin, "Equiangular navigation and guidance of a wheeled mobile robot based on range-only measurements," *Robotics and Autonomous Systems*, vol. 58, pp. 203–215, 2010.

[35] A. S. Matveev, H. Teimoori, and A. V. Savkin, "Navigation of a unicycle-like mobile robot for environmental extremum seeking," *Automatica*, vol. 47, pp. 85–91, 2011.

[36] R. Dai and C. Sun, "Path planning of spatial rigid motion with constrained attitude," *Journal of Guidance, Control, and Dynamics*, vol. 38, no. 8, pp. 1356–1365, 2015.

[37] R. Tsalik and T. Shima, "Inscribed angle guidance," *Journal of Guidance, Control, and Dynamics*, vol. 38, no. 1, pp. 30–40, 2015.

[38] Z. Chen and T. Shima, "Nonlinear optimal guidance for intercepting a stationary target," *Journal of Guidance, Control, and Dynamics*, vol. 42, no. 11, pp. 2418–2431, 2019.

[39] F. Alizadeh and D. Goldfarb, "Second-order cone programming," *Mathematical Programming*, vol. 95, no. 1, pp. 3–51, 2003.

[40] X. Liu, Z. Shen, and P. Lu, "Entry trajectory optimization by second-order cone programming," *Journal of Guidance, Control, and Dynamics*, vol. 39, no. 2, pp. 227–241, 2016.

[41] R. Hartl, S. Sethi, and R. Vickson, "A survey of the maximum principles for optimal control problems with state constraints," *SIAM Rev.*, vol. 37, no. 2, pp. 181–218, 1995.

[42] D. P. Bertsekas, A. Nedic, and A. E. Ozdaglar, *Convex Analysis and Optimization*. Belmont, Mass.: Athena Scientific, 2003, ch. 2.

Xinfu Liu received his BS degree in the Department of Automation from Central South University, Changsha, China, in 2008, and his PhD degree in Aerospace Engineering from Iowa State University, Ames, USA, in 2013. From 2013 to 2016 he was a postdoctoral research associate in Beihang University, Beijing, China. He is currently an associate professor in Aerospace Engineering at Beijing Institute of Technology. His research interests are optimal control, autonomous trajectory planning of flight vehicles, convex optimization and its applications.