UAV Collision Avoidance Using Mixed-Integer Second-Order Cone Programming

Guoxu Zhang* and Xinfu Liu†
Beijing Institute of Technology, Beijing, 100081, China

I. Introduction

With the rapid technology development in areas such as autonomous control, communication, and high-performance materials, UAVs have been increasingly applied in various missions including military reconnaissance, firefighting, field rescue, etc. [1–3]. These missions generally put forward high requirements on the autonomous decision and flight ability of UAVs. Especially, UAVs have to own onboard collision avoidance capabilities for autonomous missions, which generally requires a UAV to be able to quickly and repeatedly plan a collision-free and possibly optimal trajectory from its current position to a target position.

In recent years, many methods have been developed to plan paths or trajectories for UAVs with collision avoidance. One popular class of methods is the graph-based search methods represented by A* [4, 5], D* [6], and their variants [7–9]. These methods generally use complex spatial discretization methods to first create a rasterized map and then search collision-free paths on the map. Sampling-based methods, such as rapidly-exploring random trees (RRT) [10], have also been developed to find feasible paths, while the RRT*, an advanced RRT, is known to be able to generate optimal paths [11–13]. Since the RRT* method does not consider the UAV dynamics, a refined RRT* was developed to incorporate the UAV dynamics and generate paths without sharp turns [14]. Other methods, such as artificial potential field methods [15, 16] and spline interpolation methods [17, 18] have been widely used to solve the UAV collision avoidance problem. The collision cone method was also proposed to avoid collision with dynamic (or even deforming) objects [19, 20]. Nevertheless, these methods may be either too time-consuming for an accurate enough paths or unable to consider various practical constraints and optimization objectives.

Another popular method is to formulate the collision avoidance problem as an optimal control problem (OCP) and then solve it by convex optimization. In this method, various techniques are needed to transform the original nonconvex OCP into convex problems so that we can utilize convex optimization algorithms to efficiently solve the convex problems. Successive convex programming (SCP) is such a typical method and it can converge if certain parameters are carefully designed or adjusted [21, 22]. The SCP method was used to plan collision-free trajectories for UAVs [23, 24]. In addition, an iterative rank minimization (IRM) method, which is still based on convex optimization, was also proposed to find collision-free trajectories for UAVs [14] with guaranteed convergence. Nevertheless, it is still computationally expensive, not suitable yet for onboard implementation.

In this paper, we propose a new method that is based on successive mixed-integer second-order cone programming (SOCP) to very efficiently solve the UAV collision avoidance problem with nonlinear dynamics, and an optimization objective of minimizing the flight time is also sought. To achieve this goal, we carefully convert the original nonconvex OCP into a mixed-integer SOCP problem. A novel convexification technique we propose in this paper is to introduce a new constraint. We will see that this technique is very effective to help convexify both the nonlinear dynamics and the nonlinear objective function without losing any nonlinearity. Then, we relax the newly introduced constraint and can theoretically prove the exactness of the relaxation. In addition, the collision avoidance constraints are convexified into linear constraints with binary variables. A unique feature of our proposed convexification is that the number of binary variables is just equal to the number of obstacles. This is clearly different with the existing mixed-integer linear programming method in which the required number of binary variables is much larger than the number of obstacles [25–29]. Note that fewer binary variables are very beneficial to improve the computational efficiency of solving the associated mixed-integer problems. Finally, we can get a mixed-integer SOCP problem, which is then iteratively solved until convergence to get a solution of the original problem. The iterative algorithm does not require us to provide initial profiles for the state and control variables, and its high computational efficiency will be demonstrated by numerical examples.

To further improve the computational efficiency, our analysis shows that we can eliminate the need for iterations, or in other words, we only need to solve one mixed-integer SOCP problem. The corresponding iteration-free algorithm can find an approximate solution that is very close to the one obtained by the iterative algorithm. Moreover, it is highly efficient. For instance, numerical examples will demonstrate that it takes only around 200 ms (on a personal

*PhD Candidate, School of Aerospace Engineering; zhangguoxux@sina.com.
†Associate Professor, School of Aerospace Engineering; lau.xinfu@gmail.com. Member AIAA.

desktop) to find a minimum-time trajectory for missions with multiple obstacles.

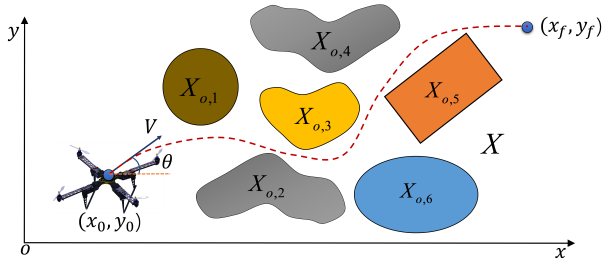

Fig. 1 A geometric view of a UAV flying in an environment with multiple obstacles.

II. Problem Formulation

Figure 1 shows a geometric view of a UAV flying from an initial position (x_0, y_0) to a target position (x_f, y_f) in an environment with multiple obstacles (the space occupied by the jth obstacle is denoted by $X_{o,j}$). The UAV has a constant speed V, and its heading angle and angular velocity are denoted as θ and w, respectively. The UAV has the following dynamics [14]

$$\begin{aligned} \dot{x} &= V\cos\theta \\ \dot{y} &= V\sin\theta \\ \dot{\theta} &= \omega \end{aligned} \tag{1}$$

The constraints that the UAV must satisfy include:

1) Control constraint: The angular velocity of a UAV is limited by
$$|\omega| \le \omega_{\max} \tag{2}$$
where $\omega_{\max}$ is the maximum allowed angular velocity.
2) Collision avoidance constraint: The UAV needs to avoid collision with all obstacles. If the number of obstacles is M, we have
$$(x, y) \in X \backslash \bigcup_{j=1}^{M} X_{o,j} \tag{3}$$
where X is the whole space and $X_{o,j}$ can represent the space occupied by an ellipse, a rectangle, or an irregularly shaped obstacle (cf. Fig. 1).
3) Terminal constraints: We can impose terminal constraints at the final time t_f as follows (note that in certain missions $\theta(t_f)$ may not be constrained)
$$x(t_f) = x_f, y(t_f) = y_f, \theta(t_f) = \theta_f \tag{4}$$

When the optimization objective is set to minimize the time of flight, we can formulate the UAV trajectory planning problem with collision avoidance as the following OCP

$$\begin{aligned} \text{Problem } O: \min\ J &= \int_0^{t_f} 1\,\mathrm{d}t \\ \text{s.t. Eqs. (1), (2), (3), and (4)} & \end{aligned} \tag{5}$$

Two mild assumptions are made on the above problem:

Assumption 1 The heading angle satisfies $\theta \in (-\pi/2, \pi/2)$.

Assumption 2 When the control constraint (2) is active in any finite interval, the collision avoidance constraint (3) is inactive in the interval.

Problem O is a nonconvex problem, and it is nonconvex due to the nonlinear dynamics and nonconvex collision avoidance constraint. In the next section, an iterative algorithm based on mixed-integer SOCP will be proposed to reliably and efficiently solve it.

Remark 1 Assumption 1 simply limits the scope of application for the proposed method in this paper. Note that the minimum-time objective will generally make the assumption easy to satisfy since the UAV will fly toward the target in order to save time. Assumption 2 is very mild, because when the control constraint (2) is active on a finite interval, the corresponding state trajectory is a circle with a radius of $V/\omega_{\max}$ (or a curvature of $\omega_{\max}/V$), and this implies that the corresponding collision avoidance constraint (3) must be inactive as long as the boundaries of all obstacles do not contain any circular arc with a curvature exactly equal to $\omega_{\max}/V$ (which is easy to satisfy).

III. Solution by Mixed-Integer Second-Order Cone Programming

In this section, we will present how to convert Problem O into an SOCP with binary variables. An iterative algorithm of successively solving the mixed-integer SOCP problems will be introduced to efficiently get a solution of Problem O. Analysis will be also given to help design an iteration-free algorithm for further improved efficiency.

A. Convexifying the Nonlinear Dynamics

Since t_f is unknown, we first choose a new independent variable with known initial and terminal values so that the dynamics can be later discretized. The state variable x can be selected since x_0 and x_f are all given, and it is also strictly monotonically increasing under Assumption 1. With x as the independent variable, the dynamics in Eq. (1)

can be equivalently rewritten as

$$\begin{aligned} y' &= \tan\theta \\ \theta' &= \frac{\omega}{V\cos\theta} \end{aligned} \tag{6}$$

where the superscript $'$ represents the derivative with respect to x.

To eliminate the nonlinearity in the y-state equation in Eq. (6), we propose to apply the change of variables $\vartheta := \tan\theta$. Then, the dynamics in Eq. (6) are equivalently transformed into

$$\begin{aligned} y' &= \vartheta \\ \vartheta' &= \sqrt{1+\vartheta^2}^3\,\omega/V \end{aligned} \tag{7}$$

where the relations $1/\cos\theta = \sqrt{(1+(\tan\theta)^2)} = \sqrt{1+\vartheta^2}$ and $\vartheta' = (\tan\theta)' = \theta'/\cos^2\theta$ are used. At the same time, with x as the independent variable and ϑ as a new state variable to replace θ, the objective function in Problem O becomes

$$J = \frac{1}{V}\int_{x_0}^{x_f} \sqrt{1+\vartheta^2}\,\mathrm{d}x \tag{8}$$

It is seen that the y-state equation in Eq. (7) is now linear, but the ϑ-state equation in Eq. (7) and the objective function in Eq. (8) are still nonlinear, mainly because the term $\sqrt{1+\vartheta^2}$ exists. Then, we propose to introduce the following new constraint

$$\delta = \sqrt{1+\vartheta^2} \tag{9}$$

so that we can replace the nonlinear term $\sqrt{1+\vartheta^2}$ by the linear term δ. Specifically, we can rewrite the dynamics in Eq. (7) as

$$\begin{aligned} y' &= \vartheta \\ \vartheta' &= \delta^3\omega/V \end{aligned} \tag{10}$$

and rewrite the objective function in Eq. (8) as the following linear one

$$\mathrm{J} = \frac{1}{V}\int_{x_0}^{x_f} \delta\,\mathrm{d}x \tag{11}$$

It should be pointed out that the introduction of the new constraint (9) can make ϑ and δ co-exist in either the dynamics or the objective function. This is in sharp contrast with the change of variables $\delta := \sqrt{1+\vartheta^2}$ which requires ϑ no longer to exist (note that the symbols "=" and ":=" have different meanings in this paper). If the change of variables is used (with the symbol ":="), the corresponding dynamics will be much more complex than the one in Eq. (10). Hence, we choose to introduce the new constraint (9). Note that this technique is very helpful to eliminate the nonlinearity in the dynamics and objective function without losing any nonlinearity.

To further convexify the dynamics in Eq. (10), we define $u := \delta^3\omega/V$. Then, the dynamics in Eq. (10) become

$$\begin{aligned} y' &= \vartheta \\ \vartheta' &= u \end{aligned} \tag{12}$$

In addition, we need to change the control constraint (2) so that the new control u is constrained, i.e.,

$$|u| \le \omega_{\max}\delta^3/V \tag{13}$$

Until now, the nonlinear dynamics in Eq. (1) have been equivalently transformed into the linear dynamics in Eq. (12) via the techniques of change of variables and the introduction of a new constraint [cf. Eq. (9)]. Correspondingly, we have the objective function in Eq. (11) and the control constraint in Eq. (13). Finally, it should be noted that the terminal constraints in Eq. (4) become

$$y(x_f) = y_f, \vartheta(x_f) = \vartheta_f \tag{14}$$

where $\vartheta_f = \tan\theta_f$.

We may notice that though we have obtained the linear dynamics, the constraints in Eqs. (9) and (13) are all nonconvex. Nevertheless, the constraint (9) can be simply relaxed with guaranteed validity, as will be seen in Sec. III.C. In addition, the constraint (13) has a unique feature that it can be rewritten as $u - \omega_{\max}\delta^3/V \le 0$ and $-u - \omega_{\max}V\delta^3/V \le 0$, where the constraint functions are all concave. We will see later in Sec. III.D that this type of constraints can be effectively convexified by successive linearization.

B. Handing the Collision Avoidance Constraints Using Binary Variables

With x as the new independent variable, the collision avoidance constraint (3) can be rewritten as

$$y \ge \bar{l}_j(x) \text{ or } y \le \underline{l}_j(x), j = 1, ..., M \tag{15}$$

where $\bar{l}_j(x) = \max\{y|(x,y) \in \partial X_{o,j}\}$, $\underline{l}_j(x) = \min\{y|(x,y) \in \partial X_{o,j}\}$, and $\partial X_{o,j}$ represents the boundary of the set $X_{o,j}$. How to get the parameters $\bar{l}_j(x)$ and $\underline{l}_j(x)$ in Eq. (15) will be discussed later in Sec. III.D. Note that the "or" constraints (15) are inconvenient to be imposed in an OCP. Considering that a UAV has two decisions to avoid collision with an obstacle, flying from either the upper side or the lower side of the obstacle, we can map these two decisions as 1 and 0 respectively. Hence, the collision avoidance constraints (15) can be equivalently transformed into

$$y \ge \bar{l}_j(x) + D(\eta_j - 1),\ y \le \underline{l}_j(x) + D\eta_j,\ j = 1, \ldots, M \tag{16}$$

where $\eta_j \in \{0, 1\}$ is a binary variable associated with the jth obstacle and $D > 0$ is a sufficiently large constant. It should be pointed out that since x is used as the independent variable, we only need M binary variables, which is equal to the number of obstacles. This proposed method is different from all previous MILP methods [25–29] in which the number of binary variables needed is much larger than the number of obstacles.

C. Relaxing the Newly Introduced Nonconvex Constraint

Based on the discussion in the preceding two subsections, we obtain a new OCP as follows

$$\begin{aligned} \text{Problem } \mathcal{E}: \min\ & J = \frac{1}{V}\int_{x_0}^{x_f} \delta \, \mathrm{d}x \\ \text{s.t. } & \text{Eqs. (9), (12), (13), (14), and (16)} \end{aligned} \tag{17}$$

It is seen that the proposed techniques for transforming Problem $\mathcal{O}$ into Problem $\mathcal{E}$ do not involve any approximation. Hence, these two problems are equivalent, stated in the following lemma

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

In Problem $\mathcal{E}$, the newly introduced constraint (9) is nonconvex. We propose to relax it into the following second-order cone constraint (which is convex)

$$\delta \geq \sqrt{1 + \vartheta^2} \tag{18}$$

With the proposed relaxation technique, a new OCP is constructed as follows

$$\begin{aligned} \text{Problem } \mathcal{R}: \min\ & J = \frac{1}{V}\int_{x_0}^{x_f} \delta \, \mathrm{d}x \\ \text{s.t. } & \text{Eqs. (18), (12), (13), (14), and (16)} \end{aligned} \tag{19}$$

In this paper, we can prove that the relaxed constraint (18) is active, given as follows

Lemma 2 Under Assumption 2, the solution of Problem $\mathcal{R}$ makes the relaxed constraint $\delta \geq \sqrt{1 + \vartheta^2}$ active almost everywhere (a.e.) in $[x_0, x_f]$.

Proof See the Appendix.

Based on Lemma 1 and 2, we have the following conclusion

Theorem 1 Under Assumptions 1-2, the solution of Problem $\mathcal{O}$ can be obtained by solving Problem $\mathcal{R}$.

Proof Denote the optimal cost of Problem $\mathcal{E}$ and Problem $\mathcal{R}$ as $J^*_{\mathcal{E}}$ and $J^*_{\mathcal{R}}$, respectively. Since the feasible set is expanded from Problem $\mathcal{E}$ to Problem $\mathcal{R}$, we have $J^*_{\mathcal{E}} \geq J^*_{\mathcal{R}}$. Lemma 2 implies that any feasible solution of Problem $\mathcal{R}$ is also feasible to Problem $\mathcal{E}$, which results in $J^*_{\mathcal{R}} \geq J^*_{\mathcal{E}}$. Hence, we get $J^*_{\mathcal{R}} = J^*_{\mathcal{E}}$, meaning that the solution of Problem $\mathcal{R}$ is also optimal to Problem $\mathcal{E}$. In addition, based on Lemma 1, the solution of Problem $\mathcal{O}$ can be obtained from the solution of Problem $\mathcal{E}$. As a result, the solution of Problem $\mathcal{O}$ can be obtained by solving Problem $\mathcal{R}$.

D. Successive Linearization and an Iterative Algorithm Based on Mixed-Integer SOCP

The successive linearization technique is applied to approximate the nonconvex control constraint (13) in Problem $\mathcal{R}$ by

$$|u| \leq \frac{\omega_{\max}}{V}[3(\delta^{(k)})^2\delta - 2(\delta^{(k)})^3] \tag{20}$$

where $\delta^{(k)}$ is the δ in the kth iteration. Next, in the discretization process if the number of discretized points is selected as $(N + 1)$, the optimization variables will include $\boldsymbol{y} = [y_0, y_1, \ldots, y_N]^T$, $\boldsymbol{\vartheta} = [\vartheta_0, \vartheta_1, \ldots, \vartheta_N]^T$, $\boldsymbol{u} = [u_0, u_1, \ldots, u_N]^T$, $\boldsymbol{\delta} = [\delta_0, \delta_1, \ldots, \delta_N]^T$, and $\boldsymbol{\eta} = [\eta_0, \eta_1, \ldots, \eta_M]^T$. Define $\boldsymbol{z} = [\boldsymbol{y}^T, \boldsymbol{\vartheta}^T, \boldsymbol{u}^T, \boldsymbol{\delta}^T]^T$. Then, Problem $\mathcal{R}$ with Eq. (13) replaced by Eq. (20) can be discretized into a mixed-integer SOCP problem with the following form

$$\begin{aligned} \text{Problem } \mathcal{D}(\boldsymbol{\delta}^{(k)}): \min_{\boldsymbol{z},\boldsymbol{\eta}}\ & \boldsymbol{c}^T\boldsymbol{z} \\ \text{s.t. } & H\boldsymbol{z} \leq \boldsymbol{p} \\ & \Theta\boldsymbol{z} - \boldsymbol{b} \succeq_K 0 \\ & g_m(\boldsymbol{z}, \boldsymbol{\eta}) \leq 0, m = 1, ..., 2M \end{aligned} \tag{21}$$

In the above problem, the linear inequality constraint is from Eqs. (12), (14), and (20). The generalized inequality constraint is from Eq. (18), which means $\Theta\boldsymbol{z} - \boldsymbol{b} \in K$, where K is the direct product of second-order cones. The last constraints represent the discretized constraints of Eq. (16). Note that when Eq. (16) is discretized, we need to compute the parameters $\bar{l}_j(x_i)$ and $\underline{l}_j(x_i)$, where x_i is the value of x at the ith discretized point. To this end, we can check the boundary of the jth obstacle at $x = x_i$ and find the maximum and minimum y-coordinates which are then set equal to $\bar{l}_j(x_i)$ and $\underline{l}_j(x_i)$, respectively.

It can be seen that Problem $\mathcal{D}(\boldsymbol{\delta}^{(k)})$ is a mixed-integer SOCP problem, and it has some parameters dependent on $\boldsymbol{\delta}^{(k)}$ [cf. Eq. (20)]. To get a solution to Problem $\mathcal{O}$, we can iteratively solve Problem $\mathcal{D}(\boldsymbol{\delta}^{(k)})$ until the sequence $\{\boldsymbol{\delta}^{(k)}\}$ converges, which is given in Algorithm 1.

Algorithm 1 An iterative algorithm for solution to Problem $\mathcal{O}$.

1) Set $k = 0$ and select an initial $\boldsymbol{\delta}$ profile $\boldsymbol{\delta}^{(0)}$.
2) In the $(k+1)$th iteration, Problem $\mathcal{D}(\boldsymbol{\delta}^{(k)})$ is solved to get $\{\boldsymbol{z}^{(k+1)}; \boldsymbol{\eta}^{(k+1)}\}$.
3) Check the convergence condition $\max\{|\boldsymbol{\delta}^{(k+1)} - \boldsymbol{\delta}^{(k)}|\} \leq \epsilon_\delta$, where ϵ_δ is a user-specified small tolerance. If it is satisfied, go to Step 4). Otherwise, set $k = k + 1$ and go back to Step 2).
4) Stop. The solution found is $\{\boldsymbol{z}^{(k+1)}; \boldsymbol{\eta}^{(k+1)}\}$.

When Algorithm 1 converges, the solution obtained is an optimal solution to Problem $\mathcal{R}$. Based on Theorem 1, this solution can be used to further get a solution to Problem $\mathcal{O}$.

Remark 2 Though it is difficult to theoretically prove the convergence of Algorithm 1, we will provide some necessary discussion. Note that almost all the proposed convexification or transformation steps do not involve any approximation and the only approximation lies in linearizing the term δ^3 in the inequality constraint (13). This brings the benefit that the convergence of the $\boldsymbol{\delta}$ profiles is generally much easier to achieve when compared to any other iterative algorithm which involves linearization on equality constraints and requires the convergence of the state (and control) profiles. In addition, our extensive numerical implementation shows that the value of $\boldsymbol{\eta}$ generally does not change since the 1st iteration. Note that when $\boldsymbol{\eta}$ does not change, we may view it as a known constant and consequently Problem $\mathcal{D}(\boldsymbol{\delta}^{(k)})$ is an SOCP problem. Iteratively solving it is guaranteed to converge since the problem is obtained from Problem $\mathcal{R}$ by just linearizing concave inequality constraints (cf. Ref. [30] for the proof). Therefore, Algorithm 1 is highly reliable.

E. An Iteration-free Algorithm

In this subsection, we will discuss how to select an appropriate initial $\boldsymbol{\delta}$ profile so that one iteration is sufficient in Algorithm 1. Recall that the iterative solution process is required in Algorithm 1 just because an error exists in linearizing Eq. (13) into Eq. (20). Nevertheless, our analysis will show that the error of the linearization is actually small when $\boldsymbol{\delta}^{(k)}$ is appropriately selected, and consequently the iterative solution process can become unnecessary. To begin with, we compute the percent error of the linearization by

$$\varepsilon = \frac{\delta^3 - [3(\delta^{(k)})^2\delta - 2(\delta^{(k)})^3]}{\delta^3} \times 100\% \tag{22}$$

Since $\delta = \sqrt{1+\vartheta^2} = 1/\cos\theta$ (note that $\vartheta := \tan\theta$), substituting $\delta = 1/\cos\theta$ into Eq. (22) yields

$$\varepsilon = \{1 - \cos^2\theta[3(\delta^{(k)})^2 - 2(\delta^{(k)})^3\cos\theta]\} \times 100\% \tag{23}$$

Figure 2 plots the values of ε with respect to θ when $\delta^{(k)}$ is set to a few different values (note that $\delta \geq 1$ since $\delta \geq \sqrt{1+\vartheta^2}$). It can be seen that when $|\theta|$ is small, the linearization error for $\delta^{(k)} = 1$ is the smallest when compared to setting $\delta^{(k)}$ as other values. Note that we can select a coordinate frame with its x-axis connecting the UAV's current position and target position so that the overall $|\theta|$ is relatively small. If $\delta^{(k)} = 1$, Fig. 2 shows that the linearization error is almost ignorable ($< 1.5\%$) when $|\theta| \leq 20$ deg, and it is still less than 15% when $|\theta|$ is as large as 40 deg. Hence, we can set the initial $\boldsymbol{\delta}$ profile $\boldsymbol{\delta}^{(0)}$ to be $[1, 1, \ldots, 1]^T$ and design an iteration-free algorithm in Algorithm 2.

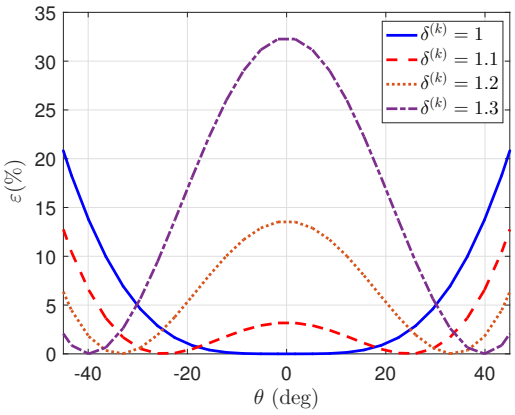

Fig. 2 The percent error of the linearization with respect to the heading angle θ for different values of $\delta^{(k)}$.

Algorithm 2 An iterative-free algorithm for solution to Problem $\mathcal{O}$.

1) Select a coordinate frame with its origin at the current position and its x-axis be ray connecting the origin and the target position. Set $\boldsymbol{\delta}^{(0)} = [1, 1, \ldots, 1]^T$.
2) Problem $\mathcal{D}(\boldsymbol{\delta}^{(0)})$ is solved to get $\{\boldsymbol{z}^{(1)}; \boldsymbol{\eta}^{(1)}\}$.
3) Stop. The solution found is $\{\boldsymbol{z}^{(1)}; \boldsymbol{\eta}^{(1)}\}$.

It should be pointed out that due to the existence of the linearization error, the solution found by Algorithm 2 may be an approximate solution to Problem $\mathcal{R}$, and based on Theorem 1 it can be used to get an approximated solution to Problem $\mathcal{O}$. Nevertheless, we will see in Sec. IV that Algorithm 2 can find solutions that are very close to those obtained by Algorithm 1.

Remark 3 The linearization error only affects the bound of the angular velocity [cf. Eq. (20)], and it may result in certain conservativeness on the available maximum angular velocity. This is also the reason why the solution obtained by Algorithm 2 may be considered as an approximate solution.

Remark 4 Figure 2 also gives some hints on how to choose $\boldsymbol{\delta}^{(0)}$ in Algorithm 1. Its value can not be too large. Otherwise, the linearization error is very large, which is likely to slow down the convergence of Algorithm 1. Roughly setting $\boldsymbol{\delta}^{(0)} = [1.1, 1.1, \ldots, 1.1]^T$ is a good choice since the linearization error is found moderate, as seen in Fig. 2. With this setting, Algorithm 1 can converge very quickly in a few iterations based on our numerical implementation.

IV. Numerical Examples

In this section, numerical examples will be shown to demonstrate the high performance of Algorithm 1 and Algorithm 2. The parameters we use include $V = 5$ m/s, $\omega_{\max} = 20$ deg/s, $N = 100$, $D = 1000$, and $\epsilon_\delta = 0.01$. The algorithms are run on a desktop with an Intel Core i5-4570 3.20 GHz and the mixed-integer SOCP problems are solved by the software ECOS [31].

A. Comparison of Algorithm 1, and Algorithm 2

We consider a UAV with $(x_0, y_0) = (0, 0)$ m and $(x_f, y_f) = (110, 0)$ m, and both $\theta(0)$ and $\theta(t_f)$ are unconstrained. There are 7 obstacles in either circular shapes or elliptic shapes (cf. Fig. 5). Those two algorithms are used to obtain a solution to Problem $\mathcal{O}$. We will also use the nonlinear programming solver SNOPT to verify the solution.

In Algorithm 1, we set $\boldsymbol{\delta}^{(0)} = [1.1, 1.1, \ldots, 1.1]^T$ (cf. Remark 4 for a reason). Figure 3 shows that it converges in just three iterations, and Fig. 4 visually shows the quick convergence of the successive $\boldsymbol{\delta}$ profiles. It is worth pointing out that the value of $\boldsymbol{\eta}$ is found to be $[0, 0, 1, 1, 0, 0, 0]^T$ in all three iterations. Note that when $\boldsymbol{\eta}$ does not change, it is expected that the algorithm can converge (cf. Remark 2). Algorithm 2, which only requires one iteration, is also used to solve the problem. Solutions are compared in Figs. 5-6, and the times of flight found by Algorithm 1 and 2 are 24.013 s and 24.016 s, respectively. Clearly, both algorithms get almost the same solution. The only small difference we can notice in the scale of the figures is that the angular velocity obtained by Algorithm 2 is slightly smaller than $\omega_{\max} = 20$ deg/s in an interval around $t = 21$ s (cf. Fig. 6), whereas in the same interval the one obtained by Algorithm 1 is equal to 20 deg/s. This phenomenon indicates that though Algorithm 2 is iteration-free, it only has slight conservativeness in the control constraint. In addition, Algorithm 2 has a computation time of 183 ms, which is much more efficient than Algorithm 1 which costs 631 ms.

We also use SNOPT to directly solve Problem $\mathcal{O}$. Note that its solution is dependent on the initial guess. When the initial guess of the trajectory is set as a straight line connecting the initial and target positions (cf. Fig. 5), the solution obtained by SNOPT (see the pink dash dotted lines in Figs. 5-6) is different with the solutions obtained by Algorithm 1 & 2, and the time of flight is longer (which is 25.275 s). Nevertheless, when the initial guess is set as two line segments shown in Fig. 5, it can be seen from Figs. 5-6 that SNOPT (which solution is the blue solid lines) verifies the optimality of the solutions obtained by Algorithm 1 & 2 since their solutions are almost the same. Note that the time of flight found by SNOPT is 24.012 s and the computation time is 1.736 s.

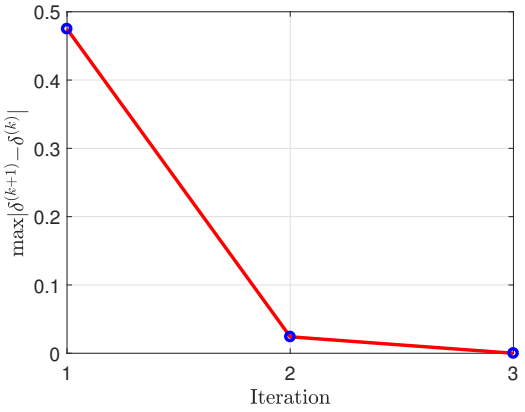

Fig. 3 The values of $\max\{|\boldsymbol{\delta}^{(k+1)} - \boldsymbol{\delta}^{(k)}|\}$ obtained in each iteration of Algorithm 1.

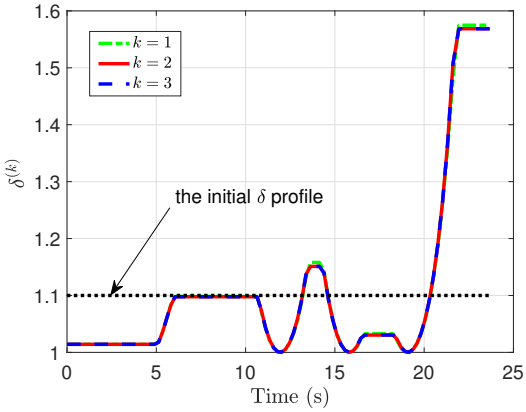

Fig. 4 The δ profiles obtained in each iteration of Algorithm 1.

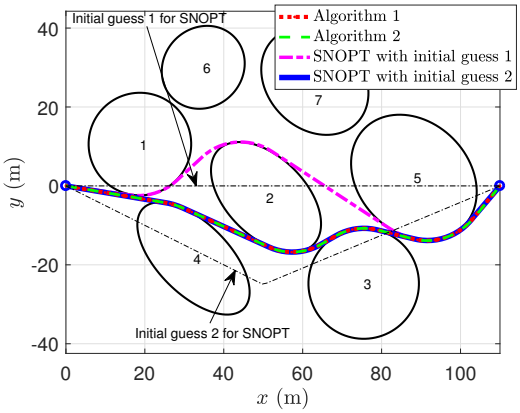

Fig. 5 Comparison of the trajectories obtained by Algorithm 1 and Algorithm 2, and verification of the solutions by using SNOPT

.

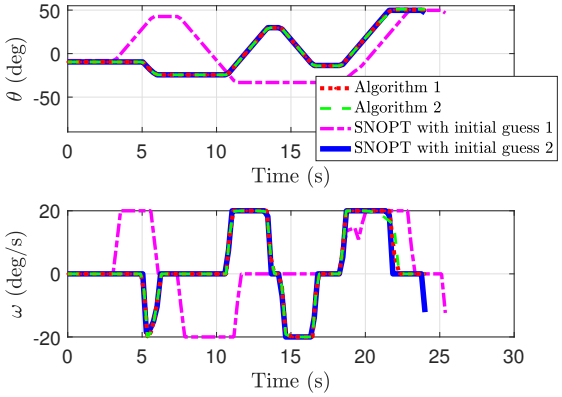

Fig. 6 Comparison of the heading angle and angular velocity profiles obtained by Algorithm 1 and Algorithm 2, and verification of the solutions by using SNOPT

.

B. Test of Algorithm 2 for Missions with Irregular Obstacles

Since Algorithm 2 is much more efficient than Algorithm 1, we further test the performance of Algorithm 2 for more missions. We set $(x_0, y_0) = (0, 0)$ m and $(x_f, y_f) = (110, 0)$ m. Eight obstacles with irregular ones are considered (cf. Fig. 7). The initial and terminal heading angles are either constrained by $\theta(0) = -45^\circ$ & $\theta(t_f) = 45^\circ$ or unconstrained, which forms two missions.

Algorithm 2 gets the solutions for the two missions with a computation time of just 217 ms and 154 ms respectively, and the times of flight found are 24.942 s and 23.549 s respectively. The solutions are plotted in Figs. 7-8. Figure 7 shows that the UAV can successfully avoid all the obstacles in both missions. In addition, it is seen from Fig. 8 that there exists slight conservativeness in the control constraint, because the maximum angular velocity is found not exactly equal to $\omega_{\max} = 20$ deg/s in some intervals. Nevertheless, the obtained trajectory in each mission is a combination of straight lines and curves with almost the minimum turn radius. Such a trajectory is well known to be able to make a UAV reach a target in the minimum time of flight. Hence, the approximate solutions obtained by Algorithm 2 are reasonable. Finally, it is worth pointing out that if the observed slight conservativeness in the control constraint is not acceptable, we can also apply Algorithm 1 for an accurate solution and it can quickly converge in three iterations for both missions.

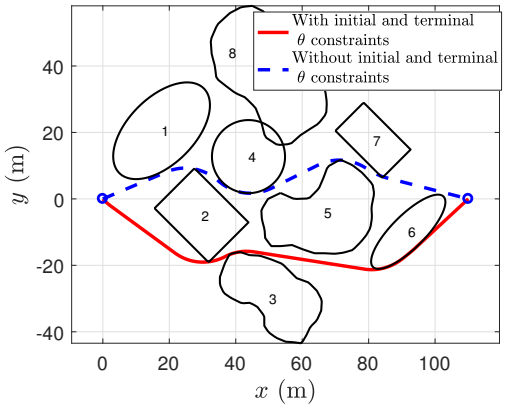

Fig. 7 Trajectories obtained by Algorithm 2 for two missions.

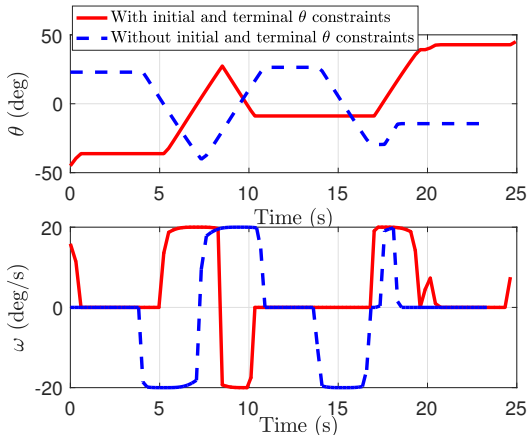

Fig. 8 Heading angle and angular velocity profiles obtained by Algorithm 2 for two missions.

V. Conclusions

The minimum-time UAV trajectory planning problem with consideration on nonlinear dynamics, bounds on angular velocity, and collision avoidance is investigated in this paper. The corresponding optimal control problem is

nonconvex. We propose to transform the problem into a mixed-integer SOCP problem and then iteratively solve the mixed-integer SOCP problems to obtain a solution of the original problem. Although different convex optimization-based methods may exist to solve the original problem, the proposed iterative algorithm offers a new way to efficiently solve the problem. In our convexification process, we do not linearize any *equality* constraints and linearization is only put on inequality constraints having *concave* constraint functions, which make the proposed iterative algorithm capable of converging quickly. To further improve the solution efficiency, we can design an iteration-free algorithm to get an approximate solution of the original problem, which means that only one mixed-integer SOCP problem needs to be solved. Numerical examples have demonstrated the effectiveness and high efficiency of those two algorithms.

It should also be highlighted that this paper proposes a novel convexification technique, which is to introduce a new constraint from defining a nonlinear term as a new variable. Note that the old variable(s) and the new variable can co-exist in the problem formulation. This technique is very useful to the specific problem in this paper since it can help eliminate the nonlinearity in both the dynamics and the objective function. Such a technique can be treated as an addition to existing convexification techniques and may find its applications in other aerospace engineering problems.

Appendix: The proof of Lemma 2

In order to apply the standard optimal control theory to Problem $\mathcal{R}$, we introduce a new variable q and equivalently rewrite Problem $\mathcal{R}$ as the following problem denoted as Problem $\bar{\mathcal{R}}$

$$\text{Problem } \bar{\mathcal{R}}: \min\ J = \frac{1}{V}\int_{x_0}^{x_f} \delta \mathrm{d}x \tag{A1}$$

$$\text{s.t. } y' = \vartheta,\ \vartheta' = u,\ \delta' = q \tag{A2}$$

$$\text{Eqs. (18), (13), (14), and (16)} \tag{A3}$$

The Hamiltonian H and Lagrangian function L for Problem $\bar{\mathcal{R}}$ are defined by $H = p_0\frac{1}{V}\delta + p_y\vartheta + p_\vartheta u + p_\delta q$ and $L = H + \lambda_1(\sqrt{1+\vartheta^2} - \delta) + \lambda_2(u - c_1\delta^3) + \lambda_3(-u - c_1\delta^3) + \lambda_4(-y + c_2) + \lambda_5(y - c_3)$, where $c_1 = \omega_{\max}/V$, $c_2 = \bar{l}_j(x) + D(\eta_j - 1)$, $c_3 = \underline{l}_j(x) + D\eta_j$, $p_0 \le 0$ is a constant, $\boldsymbol{p} = [p_y, p_\vartheta, p_\delta]^{\mathrm{T}}$ is costate vector, and $\boldsymbol{\lambda} = [\lambda_1, \lambda_2, \lambda_3, \lambda_4, \lambda_5]^{\mathrm{T}}$ is the Lagrangian multiplier. Based on the Maximum Principle [32], the necessary optimality conditions that an optimal solution has to satisfy include

1) The non-triviality condition:

$$[p_0, \boldsymbol{p}^T, \boldsymbol{\lambda}^T]^T \neq \boldsymbol{0},\ \forall x \tag{A4}$$

2) The costate differential equations:

$$p_y' = \lambda_4 - \lambda_5 \tag{A5}$$

$$p_\vartheta' = -p_y - \lambda_1\vartheta/\sqrt{1+\vartheta^2} \tag{A6}$$

$$p_\delta' = -p_0\frac{1}{V} + \lambda_1 + 3c_1\lambda_2\delta^2 + 3c_1\lambda_3\delta^2 \tag{A7}$$

3) The stationary conditions of optimality:

$$\frac{\partial L}{\partial u} = p_\vartheta + \lambda_2 - \lambda_3 = 0,\ \frac{\partial L}{\partial q} = p_\delta = 0 \tag{A8}$$

4) The complementary slackness conditions:

$$\sqrt{1+\vartheta^2} - \delta \le 0, \lambda_1 \le 0, \lambda_1(\sqrt{1+\vartheta^2} - \delta) = 0 \tag{A9}$$

$$u - c_1\delta^3 \le 0, \lambda_2 \le 0, \lambda_2(u - c_1\delta^3) = 0 \tag{A10}$$

$$-u - c_1\delta^3 \le 0, \lambda_3 \le 0, \lambda_3(-u - c_1\delta^3) = 0 \tag{A11}$$

$$-y + c_2 \le 0, \lambda_4 \le 0, \lambda_4(-y + c_2) = 0 \tag{A12}$$

$$y - c_3 \le 0, \lambda_5 \le 0, \lambda_5(y - c_3) = 0 \tag{A13}$$

In the following, we will consider two cases and prove the activeness of $\delta \ge \sqrt{1+\vartheta^2}$ in each case.

(**a**) Consider any finite interval $[x_1, x_2] \subset [x_0, x_f]$ where $|u| \le c_1\delta^3$ is not active. Assume that there exists an interval $[x_i, x_j] \subset [x_1, x_2]$ with $x_i < x_j$ such that $\delta > \sqrt{1+\vartheta^2}$. Utilizing Eqs. (A9)-(A11) yields $\lambda_1 = \lambda_2 = \lambda_3 = 0$. Then, based on Eq. (A8), we can get $p_\vartheta = p_\delta = 0$. Substituting $\lambda_1 = \lambda_2 = \lambda_3 = p_\vartheta = p_\delta = 0$ and into Eqs. (A6)-(A7) yields $p_0 = p_y = 0$.

If the collision avoidance constraints are all inactive, utilizing Eqs. (A12)-(A13) yields $\lambda_4 = \lambda_5 = 0$. If one of them is inactive, we suppose $y - c_3 \le 0$ is inactive without loss of generality. Utilizing Eq. (A13), we can get $\lambda_5 = 0$. Combining $p_y = 0$ and Eq. (A5) can generate $\lambda_4 = 0$.

In sum, $\{p_0, p_y, p_\vartheta, p_\delta, \lambda_1, \lambda_2, \lambda_3, \lambda_4, \lambda_5\} = 0$ is obtained, which clearly contradicts Eq. (A4). Hence, the assumption does no hold and $\delta \ge \sqrt{1+\vartheta^2}$ is active a.e. on the interval $[x_1, x_2]$.

(**b**) Consider any finite interval $[x_1, x_2] \subset [x_0, x_f]$ where $|u| \le c_1\delta^3$ is active. Without loss of generality, assume that $u = c_1\delta^3$ over an interval $[x_i, x_j] \subset [x_1, x_2]$. Assumption 2 implies that the collision avoidance constraints are inactive. Thus, in this interval Problem $\mathcal{R}$ becomes

$$\min\ J = \frac{1}{V}\int_{x_i}^{x_j} \delta \mathrm{d}x \tag{A14}$$

$$\text{s.t. } y' = \vartheta,\ \vartheta' = c_1\delta^3 \tag{A15}$$

$$\text{Eqs. (18) and (14)} \tag{A16}$$

The Hamiltonian for the above problem is defined by $H = p_0\frac{1}{V}\delta + p_y\vartheta + p_\vartheta c_1\delta^3$, where $p_0 \le 0$ is a constant

and $[p_y, p_\vartheta]^T$ is the costate vector. Further, we can get $\frac{\partial H}{\partial \delta} = p_0 \frac{1}{V} + 3p_\vartheta c_1 \delta^2$.

The Maximum Principle [32] reveals that the optimal δ must maximize H. Next, we will consider three situations based on p_ϑ and in each situation we will prove the activeness of $\delta \geq \sqrt{1+\vartheta^2}$.

First, we consider $p_\vartheta < 0$. This condition, together with $p_0 \leq 0$, can generate $\frac{\partial H}{\partial \delta} < 0$. Then, maximizing H with respect to δ requires δ to be its minimum. Since the the minimum of δ is $\sqrt{1+\vartheta^2}$ based on $\delta \geq \sqrt{1+\vartheta^2}$, we have $\delta = \sqrt{1+\vartheta^2}$.

Second, we consider $p_\vartheta > 0$. Since H is a cubic polynomial of δ and the cubic term has a coefficient $p_\vartheta c_1 > 0$, the optimal δ is infinity. However, $\delta = \infty$ results in $H = \infty$, which is contradictory to the fact that H must be finite if a solution exists. Hence, $p_\vartheta > 0$ can be ruled out.

Finally, we consider $p_\vartheta = 0$. In this situation, $\frac{\partial H}{\partial \delta} = p_0/V$. First, if $p_0 < 0$, $\frac{\partial H}{\partial \delta} < 0$. Hence, maximizing H yields $\delta = \sqrt{1+\vartheta^2}$. Next, if $p_0 = 0$, it is also straightforward to get $\delta = \sqrt{1+\vartheta^2}$ by contradiction (details are omitted to save space).

In summary, the above analysis shows that the relaxed constraint $\delta \geq \sqrt{1+\vartheta^2}$ is active a.e. on the $[x_0, x_f]$.

References

[1] Anderson, K., and Gaston, K. J., "Lightweight Unmanned Aerial Vehicles Will Revolutionize Spatial Ecology," *Frontiers in Ecology and the Environment*, Vol. 11, No. 3, 2013, pp. 138–146. https://doi.org/10.1890/120150.

[2] Cao, Y., "UAV Circumnavigating an Unknown Target Under a GPS Denied Environment with Range-Only Measurements," *Automatica*, Vol. 55, 2015, pp. 150–158. https://doi.org/10.1016/j.automatica.2015.03.007, .

[3] Tomic, T., Schmid, K., Lutz, P., Domel, A., Kassecker, M., Mair, E., Grixa, I. L., Ruess, F., Suppa, M., and Burschka, D., "Toward a Fully Autonomous UAV: Research Platform for Indoor and Outdoor Urban Search and Rescue," *IEEE Robotics and Automation Magazine*, Vol. 19, No. 3, 2012, pp. 46–56. https://doi.org/10.1109/mra.2012.2206473.

[4] Hart, P. E., Nilsson, N. J., and Raphael, B., "A Formal Basis for the Heuristic Determination of Minimum Cost Paths," *IEEE Transactions on Systems Science and Cybernetics*, Vol. 4, No. 2, 1968, pp. 100–107. https://doi.org/10.1109/TSSC.1968.300136.

[5] Yang, H. I., and Zhao, Y. J., "Trajectory Planning for Autonomous Aerospace Vehicles amid Known Obstacles and Conflicts," *Journal of Guidance, Control, and Dynamics*, Vol. 27, No. 6, 2004, pp. 997–1008. https://doi.org/10.2514/1.12514.

[6] Stentz, A., "Optimal and Efficient Path Planning for Partially-Known Environments," *Proceedings of the 1994 IEEE International Conference on Robotics and Automation*, Vol. 4, San Diego, CA, USA, 1994, pp. 3310–3317. https://doi.org/10.1109/ROBOT.1994.351061.

[7] Zammit, C., and Van Kampen, E.-J., "Comparison Between A* and RRT Algorithms for UAV Path Planning," *2018 AIAA Guidance, Navigation, and Control Conference*, Kissimmee, Florida, 2018. https://doi.org/10.2514/6.2018-1846.

[8] Zammit, C., and Van Kampen, E.-J., "Advancements for A* and RRT in 3D Path Planning of UAVs," *2019 AIAA Guidance, Navigation, and Control Conference*, San Diego, CA, 2019. https://doi.org/10.2514/6.2019-0920.

[9] Watanabe, T., Balci, E., and Johnson, E. N., "Least Square Sparse Mapping and Octree-based A* Algorithm," *2018 AIAA Information Systems-AIAA Infotech @ Aerospace*, Kissimmee, Florida, 2018. https://doi.org/10.2514/6.2018-2013.

[10] LaValle, S. M., Kuffner, J. J., Donald, B., et al., "Rapidly-Exploring Random Trees: Progress and Prospects," *Algorithmic and Computational Robotics: New Directions*, Vol. 5, 2001, pp. 293–308.

[11] Karaman, S., and Frazzoli, E., "Sampling-based Algorithms for Optimal Motion Planning," *The International Journal of Robotics Research*, Vol. 30, No. 7, 2011, pp. 846–894. https://doi.org/10.1177/0278364911406761.

[12] Karaman, S., Walter, M. R., Perez, A., Frazzoli, E., and Teller, S., "Anytime Motion Planning Using the RRT," *2011 IEEE International Conference on Robotics and Automation*, Shanghai, China, 2011, pp. 1478–1483. https://doi.org/10.1109/ICRA.2011.5980479.

[13] Elbanhawi, M., and Simic, M., "Sampling-based Robot Motion Planning: A Review," *IEEE Access*, Vol. 2, 2014, pp. 56–77. https://doi.org/10.1109/ACCESS.2014.2302442.

[14] Sun, C., Liu, Y., Dai, R., and Grymin, D., "Two Approaches for Path Planning of Unmanned Aerial Vehicles with Avoidance Zones," *Journal of Guidance, Control, and Dynamics*, Vol. 40, No. 8, 2017, pp. 2076–2083. https://doi.org/10.2514/1.G002314.

[15] Abdullah, E. O., and Erol, D., "Artificial Potantial Field Based Autonomus UAV Fligh in Dynamic Environment," *AIAA Aviation Technology, Integration,and Operations Conference*, Washington, D.C., 2016. https://doi.org/10.2514/6.2016-3454.

[16] Bounini, F., Gingras, D., Pollart, H., and Gruyer, D., "Modified Artificial Potential Field Method for Online Path Planning Applications," *2017 IEEE Intelligent Vehicles Symposium (IV)*, Los Angeles, 2017, pp. 180–185. https://doi.org/10.1109/IVS.2017.7995717.

[17] Jung, D., and Tsiotras, P., "On-Line Path Generation for Unmanned Aerial Vehicles Using B-Spline Path Templates," *Journal of Guidance, Control, and Dynamics*, Vol. 36, No. 6, 2013, pp. 1642–1653. https://doi.org/10.2514/1.60780.

[18] Upadhyay, S., and Ratnoo, A., "Smooth Path Planning for Unmanned Aerial Vehicles with Airspace Restrictions," *Journal of Guidance, Control, and Dynamics*, Vol. 40, No. 7, 2017, pp. 1596–1612. https://doi.org/10.2514/1.G002400.

[19] Chakravarthy, A., and Ghose, D., "Obstacle Avoidance in a Dynamic Environment: A Collision Cone Approach," *IEEE Transactions on Systems, Man, and Cybernetics*, Vol. 28, No. 5, 1998, pp. 562–574. https://doi.org/10.1109/3468.709600.

[20] Sunkara, V., Chakravarthy, A., and Ghose, D., "Collision Avoidance of Arbitrarily Shaped Deforming Objects Using Collision Cones," *IEEE Robotics and Automation Letters*, Vol. 4, No. 2, 2019, pp. 2156–2163. https://doi.org/10.1109/LRA.2019.2900535.

[21] Mao, Y., Dueri, D., Szmuk, M., and Açıkmeşe, B., "Successive Convexification of Non-Convex Optimal Control Problems with State Constraints," *IFAC-PapersOnLine*, Vol. 50, No. 1, 2017, pp. 4063–4069. https://doi.org/10.1016/j.ifacol.2017.08.789.

[22] Zhang, Z., Jin, G., and Li, J., "Penalty Boundary Sequential Convex Programming Algorithm for Non-Convex Optimal Control Problems," *ISA Transactions*, Vol. 72, 2018, pp. 229–244. https://doi.org/10.1016/j.isatra.2017.09.014.

[23] Wang, Z., Liu, L., and Long, T., "Minimum-Time Trajectory Planning for Multi-Unmanned-Aerial-Vehicle Cooperation Using Sequential Convex Programming," *Journal of Guidance, Control, and Dynamics*, Vol. 40, No. 11, 2017, pp. 2976–2982. https://doi.org/10.2514/1.G002349.

[24] Dueri, D., Mao, Y., Mian, Z., Ding, J., and Açikmeşe, B., "Trajectory Optimization with Inter-Sample Obstacle Avoidance via Successive Convexification," *2017 IEEE 56th Annual Conference on Decision and Control (CDC)*, Melbourne, VIC, Australia, 2017, pp. 1150–1156. https://doi.org/10.1109/CDC.2017.8263811.

[25] Richards, A., and How, J., "Aircraft Trajectory Planning with Collision Avoidance Using Mixed Linear Programming," *Proceedings of American Control Conference*, USA, 2002. https://doi.org/10.1109/ACC.2002.1023918.

[26] Maia, M., and Galvão, R., "On the Use of Mixed-Integer Linear Programming for Predictive Control with Avoidance Constraints," *International Journal of Robust and Nonlinear Control*, Vol. 19, No. 7, 2009, pp. 822–828. https://doi.org/10.1002/rnc.1341.

[27] Richards, A., and Turnbull, O., "Inter-Sample Avoidance in Trajectory Optimizers Using Mixed-Integer Linear Programming," *International Journal of Robust and Nonlinear Control*, Vol. 25, No. 4, 2015, pp. 521–526. https://doi.org/10.2514/6.2013-4634.

[28] Afonso, R., Galvão, R., and Kienitz, K., "Reduction in the Number of Binary Variables for Inter-Sample Avoidance in Trajectory Optimizers Using Mixed-Integer Linear Programming," *International Journal of Robust and Nonlinear Control*, Vol. 26, No. 16, 2016, pp. 3662–3669. https://doi.org/10.1002/rnc.3529.

[29] Kuwata, Y., and How, J. P., "Cooperative Distributed Robust Trajectory Optimization Using Receding Horizon MILP," *IEEE Transactions on Control Systems Technology*, Vol. 19, No. 2, 2011, pp. 423–431. https://doi.org/10.1109/TCST.2010.2045501.

[30] Liu, X., and Lu, P., "Solving Nonconvex Optimal Control Problems by Convex Optimization," *Journal of Guidance, Control, and Dynamics*, Vol. 37, No. 3, 2014, pp. 750–765. https://doi.org/10.2514/1.62110.

[31] Domahidi, A., Chu, E., and Boyd, S., "ECOS: An SOCP Solver for Embedded Systems," *2013 European Control Conference (ECC)*, IEEE, Zurich, Switzerland, 2013, pp. 3071–3076. https://doi.org/10.23919/ECC.2013.6669541.

[32] Hartl, R., Sethi, S., and Vickson, R., "A Survey of the Maximum Principles for Optimal Control Problems with State Constraints," *SIAM Review*, Vol. 37, No. 2, 1995, pp. 181–218. https://doi.org/10.1137/1037043.