# Flatness-based Quadcopter Trajectory Planning and Tracking with Continuous-time Safety Guarantees

Victor Freire and Xiangru Xu, *Member, IEEE*

***Abstract*—This work proposes a convex optimization-based framework for the trajectory planning and tracking of quadcopters that ensures continuous-time safety guarantees. Using the convexity property of B-spline curves and the differential flatness property of quadcopters, a second-order cone program is formulated to generate an optimal nominal trajectory that respects state and input constraints, including position, linear velocity, angle, angular velocity, thrust, waypoints, and obstacle avoidance constraints, rigorously in the continuous-time sense. To ensure safe trajectory tracking, a convex quadratic program is proposed based on control barrier functions, which guarantees that the actual trajectory of the quadcopter remains within a prescribed safe tube of the nominal trajectory in continuous time. Furthermore, conditions that ensure the safe tracking controller respects thrust, roll, and pitch constraints are also presented. Both the planning and control approaches are suitable for online implementation, and the effectiveness of the proposed framework is demonstrated through simulations and experiments with a Crazyflie2.1 nano quadcopter.**

***Index Terms*—Safety-critical control, quadcopters, motion planning, second-order cone program, control barrier functions**

## I. Introduction

AUTONOMOUS quadcopters have been widely used in many fields such as cinematography [1], search and rescue [2], agriculture [3] and delivery [4]. Many of these applications are safety-critical, where *safety* is defined as the rigorous satisfaction of constraints expressed in terms of states and inputs of the quadcopters. Although numerous trajectory planning and tracking methods have been proposed for quadcopters, a systematic approach that is real-time implementable and can provide a formal safety guarantee in the continuous-time sense is still largely lacking.

The trajectory planning problem is usually split into finding discrete waypoints that avoid obstacles, and generating a smooth curve that respects the system dynamics through the waypoints. However, a trajectory generated this way is normally only safe at the discrete waypoints without a formal safety guarantee in-between. The inter-sampling safety can be improved by increasing the number of samples at the expense of computation, which works well in practice but lacks a rigorous continuous-time guarantee [5]–[11]. Some recent works exploited the differential flatness property of quadcopters and the convexity property of B-spline basis functions to generate trajectories with continuous position safety guarantees [12]–[15]; however, the highly nonlinear flat maps between state/input spaces and the flat output space makes the satisfaction of many non-trivial safety constraints in the state/input space challenging, and the nonconvex optimization problems formulated in these works render the online trajectory re-planning computationally intractable.

Manuscript received 31 October 2021; revised September 6 2022; accepted 20 February 2023. Recommended by Editor in Chief Andrea Serrani. *(Corresponding author: Xiangru Xu.)*

Victor Freire and Xiangru Xu are with the Department of Mechanical Engineering, University of Wisconsin-Madison, Madison, WI 53706 USA (email: freiremelgiz@wisc.edu; xiangru.xu@wisc.edu).

Digital Object Identifier **see top of this page.**

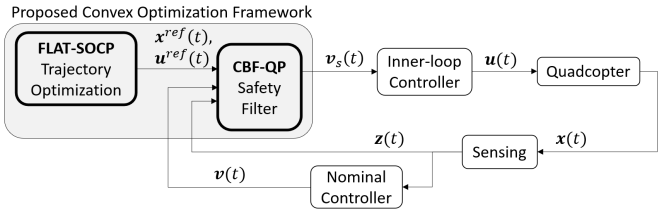



Fig. 1. Configuration of the proposed convex optimization-based planning and control framework for quadcopters with continuous-time safety guarantees. The safe trajectory planning is based on solving a second-order cone program, and the safe trajectory tracking is based on solving a quadratic program.

Some recent works aim to improve the tracking performance of quadcopters, particularly for aggressive flights [16]–[19]. Controllers designed in these works are endowed with stability and/or performance guarantees, but not safety. Safe controllers based on reinforcement learning and Gaussian Processes have been investigated for quadcopters in [20]–[22], but these works don't provide continuous safety guarantees. One line of research that can provide formal safety guarantees in the sense of forward invariance of a given safe set in continuous time is based on control barrier functions (CBFs) [23]–[27]; for example, [26] employed CBFs to guarantee visibility in quadcopter visual serving control and [27] presented a CBF-based method for assisting human pilots to teleoperate quadcopters inside constrained environments with safety guarantees.

This paper proposes a *convex optimization-based* framework that *guarantees continuous-time safety satisfaction* of quadcopters for both trajectory planning and tracking (see Figure 1). The contributions of this work are at least threefold:

1) Based on the differential flatness of quadcopters and the convexity of B-spline curves, a second-order cone program (SOCP) framework is proposed for safe trajectory *planning* for quadcopters online. The position, linear velocity, angle, angular velocity, thrust, waypoints and obstacle avoidance safety constraints of the quadcopter

TABLE I
COMPARISON OF RELEVANT QUADCOPTER TRAJECTORY GENERATION APPROACHES

| Approach | Advantages | Disadvantages |
|---|---|---|
| flatness + B-splines (this work) | formulated as convex SOCPs; continuous constraint satisfaction; position, velocity, angle, angular velocity, thrust, and waypoints constraints considered. | constraint convexification can be conservative; known safe corridor required in collision avoidance. |
| flatness + B-splines [13] | continuous constraint satisfaction; waypoint, thrust and angle constraints considered. | formulated as nonconvex problem. |
| flatness + piecewise polynomials [6], [7] | formulated as convex QPs; position, velocity, acceleration and input constraints considered. | constraints satisfied only at discrete points. |
| flatness + Bernstein polynomials [28] | continuous constraint satisfaction when convergence achieved; admit general constraints. | optimization can be nonconvex; convergence in the limit when the polynomial degree approaches infinity. |
| optimal control [29], [30] | admits general state-space constraints; generic solvers exist. | nonconvex problem in general; constraints satisfied only at discrete points. |
| primitives [31] | very fast; angular velocity, thrust, and translational constraints considered. | no constraint satisfaction guarantee; verification must be done offline. |
| successive convexification [8] | formulated as convex problems, inter-sample collision avoidance satisfied. | can only handle simple obstacle geometries; locally optimal trajectories. |

are relaxed as second-order cone constraints in the SOCP, and the trajectory generated by solving the SOCP is proven to rigorously satisfy all the safety constraints in the continuous-time sense.

2) A control barrier function (CBF)-based quadratic program (QP) method is proposed to generate safe trajectory *tracking* controllers, such that the real trajectory of the quadcopter lies in a prescribed safe tube of the nominal trajectory in the continuous-time sense. Sufficient conditions that guarantee the rigorous satisfaction of the thrust, roll angle and pitch angle constraints are also provided.
3) Simulations and experiments are presented to demonstrate the effectiveness and computational tractability of the proposed approach.

The remainder of the paper is organized as follows: Sec. II reviews related works; Sec. III introduces necessary preliminaries; Sec. IV provides sufficient conditions for continuous-time set inclusion using B-spline curves; Sec. V provides the SOCP-based framework for quadcopter trajectory planning with rigorous continuous-time safety guarantees; Sec. VI presents a CBF-QP-based method for safe trajectory tracking; Sec. VII presents the simulations and experiments and Sec. VIII concludes the paper.

## II. RELATED WORK

While a large number of trajectory planning methods for quadcopters have been proposed in the literature, an approach that is both real-time implementable and capable of providing rigorous safety guarantees in continuous-time is still lacking. Most relevant to this work are the methods based on differential flatness [32]. In the seminal work [6], the authors showed that quadcopter systems are differentially flat, and they proposed a convex QP-based method to synthesizing three-dimensional trajectories (parameterized as piecewise polynomials) for quadcopters with constraints on positions, velocities, accelerations and inputs; however, the trajectories generated may violate the constraints as they are only enforced at certain discrete time instances. This method was extended in [7], where the authors formulated the trajectory generation problem as a numerically stable unconstrained QP that is also coupled with a kinematic planner; although this approach is computationally more efficient, it provides no improvement on constraint satisfaction. In [28], the authors proposed a discrete-time approximation of continuous-time optimal control problems for differentially flat systems by using Bernstein polynomials; however, the convergence of the discrete solution to the solution of the continuous-time problem requires the Bernstein polynomial degree to approach infinity. In [12] the authors considered the constrained trajectory generation problem of a 2D 3-DOF model of a fixed-wing UAV and formulated it as an optimization problem by using the B-splines parametrization; although the trajectory generated ensures continuous validation of the state and input constraints, the resulting optimization problem is nonconvex and difficult to implement online. Methods based on the B-spline parametrization of trajectories for quadcopters were also explored in works such as [13], which formulated a nonconvex optimization problem with waypoints, thrust and angle constraints, and [15], which provided a computationally efficient treatment of obstacle avoidance and considered velocity and acceleration constraints, but fails to address other important state/input constraints and lacks rigorous safety analysis. Our method also builds on B-spline curves and differential flatness, but the quadcopter trajectory planning problem in our framework is formulated as a convex SOCP with rigorous continuous satisfaction for position, linear velocity, angle, angular velocity, and thrust constraints.

Besides flatness-based methods, various other approaches for quadcopter trajectory planning have been proposed in the literature [9], [29], [31], [33]–[35]. In [31] the authors proposed a computationally efficient, motion primitives-based trajectory generation approach, but their method doesn't admit constraint satisfaction at solve time. In [35] the authors proposed a hybrid approach that first used sampling-based techniques to find an obstacle-free path and then used convex optimization to obtain a feasible and safe trajectory; however, this approach provides no continuous safety guarantees. In [8] the authors proposed a successive convexification approach to converge towards obstacle-free trajectories in the continuous-time sense;

however, their method relies on finite violations derived from polynomial dynamics systems. A comparison of the advantages and disadvantages of relevant approaches for quadcopter trajectory generation is given in Table I.

## III. Preliminaries

### A. Quadcopter Dynamics & Differential Flatness

We consider the following 9-state quadcopter model that neglects aerodynamic effects and external disturbances (e.g., wind) in the trajectory planning problem [6]:

$$\ddot{\mathbf{r}} = T\mathbf{z}_B - g\mathbf{z}_W, \tag{1a}$$

$$\dot{\boldsymbol{\xi}} = N^{-1}(\boldsymbol{\xi})\boldsymbol{\omega}, \tag{1b}$$

$$N(\boldsymbol{\xi}) = \begin{bmatrix} 1 & 0 & -\sin\theta \\ 0 & \cos\phi & \sin\phi\cos\theta \\ 0 & -\sin\phi & \cos\phi\cos\theta \end{bmatrix}, \tag{1c}$$

$$R_B^W = \begin{bmatrix} c\theta c\psi & s\phi s\theta c\psi - c\phi s\psi & s\phi s\psi + c\phi s\theta c\psi \\ c\theta s\psi & s\phi s\theta s\psi + c\phi c\psi & c\phi s\theta s\psi - s\phi c\psi \\ -s\theta & s\phi c\theta & c\phi c\theta \end{bmatrix} = \begin{bmatrix} \mathbf{x}_B & \mathbf{y}_B & \mathbf{z}_B \end{bmatrix}, \tag{1d}$$

where $\mathbf{z}_W = (0,0,1)^T$ is the world-frame's z-axis, $\mathbf{r} = (x,y,z)^T$ is the position vector, $\boldsymbol{\xi} = (\phi,\theta,\psi)^T$ is the Z-Y-X Euler angle vector (see Figure 2), $\boldsymbol{\omega} = (p,q,r)^T$ is the angular velocity vector, $T$ is the normalized thrust, and $s$, $c$ stand for sin and cos, respectively. The state and input vectors are expressed, respectively, as:

$$\mathbf{x} = \begin{bmatrix} \mathbf{r}^T & \boldsymbol{\xi}^T & \dot{\mathbf{r}}^T \end{bmatrix}^T, \quad \mathbf{u} = \begin{bmatrix} T & \boldsymbol{\omega}^T \end{bmatrix}^T. \tag{2}$$

The quadcopter model shown in (1) is known to be differentially flat [6], [37]. By choosing flat outputs as $\boldsymbol{\sigma} = (x,y,z,\psi)^T$, the state and input can be expressed as a function of $\boldsymbol{\sigma}$ and its derivatives for some $\Phi$:

$$(\mathbf{x},\mathbf{u}) = \Phi(\boldsymbol{\sigma},\dot{\boldsymbol{\sigma}},\ddot{\boldsymbol{\sigma}},\dddot{\boldsymbol{\sigma}}), \tag{3}$$

where the expressions of $\Phi$ are given as [6], [38]:

$$\phi = \sin^{-1}\left(\frac{\mathbf{z}_W^T\mathbf{y}_B}{\cos(\sin^{-1}(\mathbf{z}_W^T\mathbf{x}_B))}\right), \qquad p = -\mathbf{y}_B\cdot\mathbf{h}_\omega,$$

$$\theta = -\sin^{-1}(\mathbf{z}_W^T\mathbf{x}_B), \qquad q = \mathbf{x}_B\cdot\mathbf{h}_\omega,$$

$$T = \sqrt{\ddot{x}^2+\ddot{y}^2+(\ddot{z}+g)^2}, \qquad r = \mathbf{z}_B\cdot\dot{\psi}\mathbf{z}_W,$$

with intermediate quantities:

$$\mathbf{T} = \begin{bmatrix} \ddot{x} & \ddot{y} & \ddot{z}+g \end{bmatrix}^T, \quad \mathbf{y}_C = \begin{bmatrix} -\sin\psi & \cos\psi & 0 \end{bmatrix}^T,$$

$$\mathbf{x}_B = \frac{\mathbf{y}_C\times\mathbf{z}_B}{\|\mathbf{y}_C\times\mathbf{z}_B\|_2}, \quad \mathbf{y}_B = \mathbf{z}_B\times\mathbf{x}_B, \quad \mathbf{z}_B = \frac{\mathbf{T}}{\|\mathbf{T}\|_2},$$

$$\mathbf{h}_\omega = \left(\dddot{\mathbf{r}} - (\mathbf{z}_B\cdot\dddot{\mathbf{r}})\mathbf{z}_B\right)/\|\mathbf{T}\|_2.$$

Generating a trajectory for differentially flat systems reduces to finding a sufficiently smooth flat output trajectory [32]. In the case of system (1), the flat trajectory $\boldsymbol{\sigma}(t)$ needs to be at least thrice-differentiable.

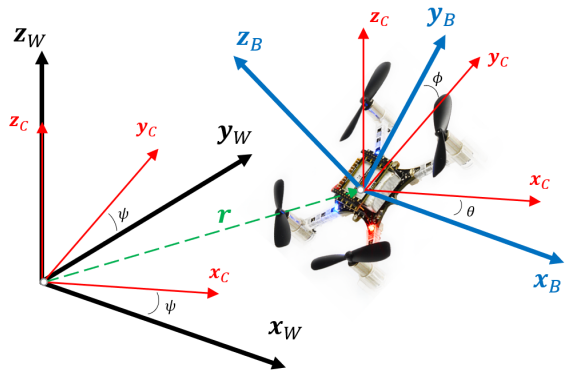



Fig. 2. Inertial (or world) coordinate frame $W$ (black) and body coordinate frame $B$ (blue). The intermediate coordinate frame $C$ (red) is also shown. Crazyflie2.1 figure from [36].

### B. B-Spline Curves

B-splines are widely used to parameterize trajectories because of their nice properties and ease of use. For example, the smoothness of B-splines makes the differentiability constraint of the flat trajectory $\boldsymbol{\sigma}(t)$ easy to satisfy and their convexity properties allow one to explore state and input constraints in continuous time.

Given a *knot vector* $\boldsymbol{\tau} = (\tau_0,\ldots,\tau_v)^T$ satisfying $\tau_i \le \tau_{i+1}$ where $i = 0,\ldots,v-1$, a $d$-th degree *B-spline basis* with $d \in \mathbb{Z}_{>0}$ is computed recursively as [39]:

$$\lambda_{i,0}(t) = \begin{cases} 1, & \tau_i \le t < \tau_{i+1}, \\ 0, & \text{otherwise}, \end{cases}$$

$$\lambda_{i,d}(t) = \frac{t-\tau_i}{\tau_{i+d}-\tau_i}\lambda_{i,d-1}(t) + \frac{\tau_{i+d+1}-t}{\tau_{i+d+1}-\tau_{i+1}}\lambda_{i+1,d-1}(t).$$

In this work, we consider the *clamped, uniform B-spline basis*, which is defined over knot vectors satisfying:

$$\text{(clamped)} \quad \tau_0 = \ldots = \tau_d, \quad \tau_{v-d} = \ldots = \tau_v, \tag{4a}$$

$$\text{(uniform)} \quad \tau_{d+1} - \tau_d = \ldots = \tau_{N+1} - \tau_N, \tag{4b}$$

where $N = v-d-1$. A $d$-th degree *B-spline curve* $\mathbf{s}(t)$ is a $m$-dimensional parametric curve built by linearly combining *control points* $\mathbf{P}_i \in \mathbb{R}^m (i = 0,\ldots,N)$ and B-spline basis functions of the same degree. Noting $\mathbf{s}^{(0)}(t) = \mathbf{s}(t)$, we generate a B-spline curve and its $r$-th order derivative by:

$$\mathbf{s}^{(r)}(t) = \sum_{i=0}^{N} \mathbf{P}_i \mathbf{b}_{r,i+1}^T \boldsymbol{\Lambda}_{d-r}(t) = PB_r\boldsymbol{\Lambda}_{d-r}(t), \tag{5}$$

where the control points are grouped into a matrix

$$P = \begin{bmatrix} \mathbf{P}_0 & \ldots & \mathbf{P}_N \end{bmatrix} \in \mathbb{R}^{m\times(N+1)}, \tag{6}$$

the basis functions are grouped into a vector

$$\boldsymbol{\Lambda}_{d-r}(t) = [\lambda_{0,d-r}(t),\ldots,\lambda_{N+r,d-r}(t)]^T \in \mathbb{R}^{N+r+1}, \tag{7}$$

and $\mathbf{b}_{r,i}^T$ is the $i$-th row of a time-invariant matrix $B_r$ whose construction is given in Appendix A.

**Definition 1.** *The columns of* $P^{(r)} \triangleq PB_r$ *are called the* $r$*-th order virtual control points (VCPs) of* $\mathbf{s}(t)$ *and denoted as* $\mathbf{P}_i^{(r)}$ *where* $i = 0,1,\ldots,N+r$*, i.e.,*

$$P^{(r)} = PB_r = \begin{bmatrix} \mathbf{P}_0^{(r)} & \ldots & \mathbf{P}_{N+r}^{(r)} \end{bmatrix}. \tag{8}$$

Note that $P^{(0)} = P$ and that the first and last $r$ columns of $P^{(r)}$ are zero vectors because of the form of $B_r$.

Some properties of B-splines that will be used in the following sections are listed below (see [39], [40] for more details).

P1) *Continuity*. Given a clamped knot vector satisfying (4a), $\mathbf{s}(t)$ is $C^\infty$ for any $t \notin \boldsymbol{\tau}$ and $C^{d-1}$ for any $t \in \boldsymbol{\tau}$.
P2) *Convexity*. The B-Spline basis functions have the "partition of unity" property, meaning that: (i) $\lambda_{i,d}(t) \geq 0$ and (ii) $\sum_{i=0}^{N} \lambda_{i,d}(t) = 1,\ \forall t \in [\tau_0, \tau_v)$.
P3) *Local Support*. Any B-spline basis is only nonzero over a local set of knots: $\lambda_{i,d}(t) \neq 0$ iff $t \in [\tau_i, \tau_{i+d+1})$.
P4) *Strong Convexity*. Segments of the B-spline curve are contained within the convex hull of a limited set of control points. Combining each segment we obtain $\mathbf{s}(t) \in \bigcup_{i=d}^{N} \text{Conv}\{\mathbf{P}_{i-d}, \ldots, \mathbf{P}_i\}$.

### C. Control Barrier Functions

The concept of zeroing control barrier functions (CBFs) was first proposed in [24], and has proven to be a powerful control design method to ensure the safety (in the sense of controlled invariance) of control affine systems [23], [41]. A provably safe control law is generated by solving a CBF-QP online [24]. In this work, we will use time-varying CBFs with relative degree 2 and their control sharing property [42].

Consider a time-varying affine control system

$$\dot{\mathbf{x}} = f(t, \mathbf{x}) + g(t, \mathbf{x})u, \tag{9}$$

where $\mathbf{x} \in \mathbb{R}^n$, $u \in U \subset \mathbb{R}$ and $f : \mathbb{R} \times \mathbb{R}^n \to \mathbb{R}^n$, $g : \mathbb{R} \times \mathbb{R}^n \to \mathbb{R}^n$ are piecewise continuous in $t$ and locally Lipschitz in $\mathbf{x}$. Given a sufficiently smooth time-varying function $h(t, \mathbf{x}) : \mathbb{R} \times \mathbb{R}^n \to \mathbb{R}$, we define the modified Lie derivative of $h(t, \mathbf{x})$ along $f$ as $\bar{L}_f^i h \triangleq (\frac{\partial}{\partial t} + L_f)^i h$ where $i$ is a non-negative integer.

**Definition 2.** [42] *Given a control system* (9)*, a $C^r$ function $h(t, \mathbf{x}) : \mathbb{R} \times \mathbb{R}^n \to \mathbb{R}$ with a relative degree $r$ is called a CBF if there exists a column vector $\mathbf{a} = (a_1, \ldots, a_r)^T \in \mathbb{R}^r$ such that for all $\mathbf{x} \in \mathbb{R}^n$, $t \geq 0$,*

$$\sup_{u \in U} \Big[ L_g \bar{L}_f^{r-1} h(t, \mathbf{x})u + \bar{L}_f^r h(t, \mathbf{x}) + \mathbf{a}^T \eta(t, \mathbf{x}) \Big] \geq 0, \tag{10}$$

*where $\eta(t, \mathbf{x}) = (\bar{L}_f^{r-1} h, \bar{L}_f^{r-2} h, \ldots, h)^T \in \mathbb{R}^r$, and the roots of $p_0^r(\lambda) = \lambda^r + a_1 \lambda^{r-1} + \ldots + a_{r-1}\lambda + a_r$ are negative reals $-\lambda_1, \ldots, -\lambda_r < 0$.*

Define a series of functions:

$$\nu_0(t, \mathbf{x}) = h(t, \mathbf{x}),\ \nu_k = \Big( \frac{\mathrm{d}}{\mathrm{d}t} + \lambda_k \Big) \nu_{k-1},\ 1 \leq k \leq r. \tag{11}$$

For any $t \geq 0, \mathbf{x} \in \mathbb{R}^n$, define the set of admissible inputs satisfying the CBF condition as $\mathcal{K}_h(t, \mathbf{x}) \triangleq \{u \in U \mid L_g \bar{L}_f^{r-1} h(t, \mathbf{x})u + \bar{L}_f^r h(t, \mathbf{x}) + \mathbf{a}^T \eta(t, \mathbf{x}) \geq 0\}$.

**Lemma 1.** [42] *Given a control system* (9)*, if 1) $h(t, \mathbf{x})$ with a relative degree $r$ is a CBF such that* (10) *holds and the roots of $p_0^r(\lambda) = \lambda^r + a_1 \lambda^{r-1} + \ldots + a_{r-1}\lambda + a_r$ are $-\lambda_1, \ldots, -\lambda_r < 0$, 2) $\nu_i$ defined in* (11) *satisfy $\nu_i(0, \mathbf{x}_0) \geq 0$ for $i = 0, 1, \ldots, r-1$, then any controller $u(t, \mathbf{x}) \in \mathcal{K}_h(t, \mathbf{x})$ that is Lipschitz in $\mathbf{x}$ will render $h(t, \mathbf{x}) \geq 0$ for any $t \geq 0$.*

Given a control system (9) and $q (q \geq 2)$ CBFs $h_i(t, \mathbf{x}), i = 1, \ldots, q$, the *control-sharing property* of CBFs was proposed in [42] to ensure the feasibility of the CBF-QP that includes multiple CBF constraints. We refer the reader to Definition 2 in [42] for more detail.

### D. Second-order Cone Constraints

A *second-order cone program* (SOCP) is a type of convex optimization problem of the following form [43], [44]:

$$\begin{aligned} \text{min.} \quad & \mathbf{f}^T \mathbf{x} \\ \text{s. t.} \quad & \|A_i \mathbf{x} + \mathbf{b}_i\|_2 \leq \mathbf{c}_i^T \mathbf{x} + d_i, \quad i = 1, \ldots, m, \end{aligned}$$

where $\mathbf{x} \in \mathbb{R}^n$ is the optimization variable, and the problem parameters are $\mathbf{f} \in \mathbb{R}^n$, $A_i \in \mathbb{R}^{(n_i - 1) \times n}$, $\mathbf{b}_i \in \mathbb{R}^{n_i - 1}$, $\mathbf{c}_i \in \mathbb{R}^n$ and $d_i \in \mathbb{R}$. Constraints of the form: $\|A\mathbf{x} + \mathbf{b}\|_2 \leq \mathbf{c}^T \mathbf{x} + d$ are called *second-order cone* (SOC) constraints of dimension $n$. Many commonly-seen convex constraints can be formulated as SOC; for example, ellipsoidal constraints ($\mathbf{x}^T Q \mathbf{x} \leq r,\ Q \succeq 0$) and polyhedral constraints ($A\mathbf{x} \leq \mathbf{b}$) [44]. SOCP encompasses many types of important convex programs as special cases, such as linear programs, convex QPs and convex quadratically constrained QPs. SOCP can be solved in polynomial time via the interior-point method and off-the-shelf solvers such as MOSEK exist [45].

## IV. Continuous-Time Set Inclusion of B-spline Curves & Their Derivatives

In this section, continuous-time convex set inclusion of B-spline curves and their derivatives will be considered. These results provide an intuitive description of the strong convexity property of B-spline curves and their derivatives, which is the key to enforcing the continuous-time state and input constraint satisfaction of quadcopter trajectories.

**Lemma 2.** *The $r$-th derivative of a clamped B-spline curve defined as in* (5) *is contained within the convex hull of its $r$-th order virtual control points such that:*

$$\mathbf{s}^{(r)}(t) \in \textit{Conv}\{\mathbf{P}_0^{(r)}, \ldots, \mathbf{P}_{N+r}^{(r)}\}, \quad t \in [\tau_0, \tau_v). \tag{12}$$

*Furthermore, for $d \leq i \leq N$,*

$$\mathbf{s}^{(r)}(t) \in \textit{Conv}\{\mathbf{P}_{i-d+r}^{(r)}, \ldots, \mathbf{P}_i^{(r)}\}, \quad t \in [\tau_i, \tau_{i+1}). \tag{13}$$

*By* (13)*, condition* (12) *can be reduced to:*

$$\mathbf{s}^{(r)}(t) \in \textit{Conv}\{\mathbf{P}_r^{(r)}, \ldots, \mathbf{P}_N^{(r)}\}, \quad t \in [\tau_0, \tau_v). \tag{14}$$

*Proof.* Applying the definition of VCP to (5) we obtain $\mathbf{s}^{(r)}(t) = P^{(r)} \boldsymbol{\Lambda}_{d-r}(t) = \sum_{i=0}^{N+r} \lambda_{i,d-r}(t) \mathbf{P}_i^{(r)}$. Applying P2) of B-splines shown in Section III-B, we determine that $\mathbf{s}^{(r)}(t)$ is a convex combination of $r$-th order VCPs,

which is equivalent to (12). Using P3) of B-splines we have $\sum_{j=i-d+r}^{i} \lambda_{j,d-r}(t) = 1, t \in [\tau_i, \tau_{i+1}), d \leq i \leq N$. Leveraging this result, we can also write (5) as $\mathbf{s}^{(r)}(t) = \sum_{j=i-d+r}^{i} \lambda_{j,d-r}(t)\mathbf{P}_j^{(r)}, \forall t \in [\tau_i, \tau_{i+1}), d \leq i \leq N$, which is equivalent to (13). Taking the union of all nonempty knot segments in the knot vector $\boldsymbol{\tau}$ results in (14). □

**Proposition 1.** *Given a convex set $\mathcal{S}$ and the $r$-th derivative of a clamped B-spline curve $\mathbf{s}^{(r)}(t)$ defined as in* (5)*, if*

$$\mathbf{P}_j^{(r)} \in \mathcal{S}, \quad j = r, \ldots, N \tag{15}$$

*holds, then $\mathbf{s}^{(r)}(t) \in \mathcal{S},\ t \in [\tau_0, \tau_v)$. If*

$$\mathbf{P}_j^{(r)} \in S, \quad j = i - d + r, \ldots, i, \quad i \in \{d, \ldots, N\} \tag{16}$$

*holds, then $\mathbf{s}^{(r)}(t) \in \mathcal{S},\ t \in [\tau_i, \tau_{i+1})$.*

*Proof.* From (14) and the convexity property, it follows that containing $\mathbf{s}^{(r)}(t)$ in a convex set $\mathcal{S}$ for $t \in [\tau_0, \tau_v)$ can be achieved by containing $\mathbf{P}_j^{(r)}$ in $\mathcal{S}$ for $j = r, \ldots, N$. This results in (15). Furthermore, containing $\mathbf{s}^{(r)}(t)$ in $\mathcal{S}$ for a nonempty time interval $[\tau_i, \tau_{i+1})$ reduces to containing $d-r+1$ $r$th order VCPs in $\mathcal{S}$ (i.e., those that form the convex hull in (13)). This is equivalent with (16). □

**Remark 1.** *Proposition 1 above generalizes Proposition 1 in [12] from an interval set inclusion to a more general convex set inclusion. This will be key when considering spherical and conic sets to ensure the safety of quadcopters as will be shown in the following sections.*

## V. Trajectory Planning with Continuous-Time Safety Guarantee

This section presents a SOCP framework to solve a safe trajectory planning problem that includes various state and input safety constraints, such that the reference trajectory satisfies the safety constraints rigorously in the continuous-time sense. Since the flat output $\psi$ is irrelevant to the state and input constraints considered in this paper, we set it to zero and limit ourselves to finding a sufficiently smooth, position flat output trajectory defined as $\mathbf{r}(t) = (x(t), y(t), z(t))^T$.

### A. Encoding Safety Specifications as SOC Constraints

Consider the following state and input safety constraints that must be satisfied rigorously for all $t \in [\tau_0, \tau_v)$:

$$\mathbf{x}(t) \in \mathcal{X}_p \cap \mathcal{X}_v \cap \mathcal{X}_\xi, \quad \mathbf{u}(t) \in \mathcal{U}_T \cap \mathcal{U}_\omega, \tag{17}$$

where the state $\mathbf{x}(t)$ and input $\mathbf{u}(t)$ are defined in (2), and $\mathcal{X}_p, \mathcal{X}_v, \mathcal{X}_\xi, \mathcal{U}_T, \mathcal{U}_\omega$ are the constraint sets for the position, linear velocity, roll and pitch angles, total thrust, and angular velocity, respectively. The procedure will be to formulate each constraint in flat space and then use Proposition 1 to derive sufficient conditions for (17). In Section V-C, we will discuss how to use Proposition 1 to achieve obstacle avoidance by satisfying local position constraints (called "safe corridors") over sub-intervals of $[\tau_0, \tau_v]$. Apart from (17), we will also consider waypoint constraints that have to be satisfied at certain discrete times.

*1) Position Constraints $\mathbf{x} \in \mathcal{X}_p$:* The position constraint set $\mathcal{X}_p$ encodes limited flight volumes as follows:

$$\mathcal{X}_p = \{\mathbf{x} \in \mathbb{R}^9 \mid \mathbf{r} \in \mathcal{K}_p\}, \tag{18}$$

where $\mathcal{K}_p$ is a given SOC. Proposition 1 immediately provides sufficient conditions for $\mathbf{x}(t) \in \mathcal{X}_p,\ t \in [\tau_0, \tau_v)$:

$$\mathbf{P}_j \in \mathcal{K}_p, \quad j = 0, \ldots, N. \tag{19}$$

*2) Linear Velocity Constraints $\mathbf{x} \in \mathcal{X}_v$:* Limiting flight velocities can reduce the risk of injury or equipment damage. Consider the linear velocity constraint set defined as:

$$\mathcal{X}_v = \{\mathbf{x} \in \mathbb{R}^9 : \|\dot{\mathbf{r}}\|_2 \leq \overline{v}\}, \tag{20}$$

where $\overline{v} \geq 0$ denotes the bound of the maximal speed. Using Proposition 1, the constraint $\mathbf{x}(t) \in \mathcal{X}_v,\ t \in [\tau_0, \tau_v)$ can be formulated as a constraint on the 1-st order VCPs:

$$\left\|\mathbf{P}_j^{(1)}\right\|_2 \leq \overline{v}, \quad j = 1, \ldots, N. \tag{21}$$

*3) Angular Constraints $\mathbf{x} \in \mathcal{X}_\xi$:* Providing bounds for the roll and pitch angles can enhance system stability and improve the accuracy of the dynamics model (1) by limiting aerodynamic effects. This is especially relevant when implementing controllers that leverage linearizations (e.g., LQR or linear MPC) or small-angle approximations. Consider the angular constraints set defined as follows:

$$\mathcal{X}_\xi = \{\mathbf{x} \in \mathbb{R}^9 : |\phi|, |\theta| \leq \epsilon\}, \tag{22}$$

where $\epsilon < \pi/2$ is a given bound for the roll and pitch angles. We begin by exposing the geometric meaning and convexity of a result on quadcopter differential flatness from [46]. The following lemma states that if the acceleration is contained in a cone, then the roll and pitch angles are bounded regardless of the yaw angle.

**Lemma 3.** *Given a positive scalar $0 < \epsilon < \pi/2$, and the following SOC:*

$$\mathcal{K}_\epsilon \triangleq \{\mathbf{p} \in \mathbb{R}^3 : \|A_\epsilon \mathbf{p}\|_2 \leq p_3 + g\}, \tag{23}$$

*where $A_\epsilon = diag(\cot\epsilon, \cot\epsilon, 0)$, if the acceleration $\ddot{\mathbf{r}}(t) \in \mathcal{K}_\epsilon$, then $|\phi(t)|, |\theta(t)| \leq \epsilon$ regardless of the yaw angle $\psi(t)$.*

*Proof.* Let $\ddot{\mathbf{r}}(t) \in \mathcal{K}_\epsilon$ and expand the SOC (23) to arrive at $\ddot{x}^2(t)\cot^2\epsilon + \ddot{y}^2(t)\cot^2\epsilon \leq (\ddot{z}(t) + g)^2$. Let $k_1(t) \triangleq \ddot{x}(t)$, $k_2(t) \triangleq \ddot{y}(t)$ and $k_3(t) \triangleq \ddot{z}(t) + g$. Dividing the inequality above by $\cot^2\epsilon \neq 0$ yields $\big(k_1^2(t) + k_2^2(t)\big)/k_3^2(t) \leq \tan^2\epsilon$. Then, the conclusion follows from Proposition 2 in [46]. □

Using Lemma 3, the following result provides sufficient conditions that are convex on the 2nd order VCPs $\mathbf{P}_i^{(2)}$, such that the angular constraint $\mathbf{x}(t) \in \mathcal{X}_\xi$ is satisfied for $t \in [\tau_0, \tau_v)$.

**Lemma 4.** *Consider B-spline curve $\mathbf{r}(t)$ defined as in* (5)*. If*

$$\left\|A_\epsilon \mathbf{P}_j^{(2)}\right\|_2 \leq \mathbf{z}_W^T \mathbf{P}_j^{(2)} + g, \quad j = 2, \ldots, N, \tag{24}$$

*holds with $A_\epsilon$ given in Lemma 3, then $\mathbf{x}(t) \in \mathcal{X}_\xi, \forall t \in [\tau_0, \tau_v)$.*

*Proof.* By Proposition 1, we have:

$$\left\|A_\epsilon \mathbf{r}^{(2)}(t)\right\|_2 \leq \mathbf{z}_W^T \mathbf{r}^{(2)}(t) + g, \quad t \in [\tau_0, \tau_v). \tag{25}$$

Since condition (25) is equivalent to $\ddot{\mathbf{r}}(t) \in \mathcal{K}_\epsilon$ with $\mathcal{K}_\epsilon$ given as in (23), it follows from Lemma 3 that $|\phi(t)|, |\theta(t)| \leq \epsilon$, $\forall t \in [\tau_0, \tau_v)$, which means that $\mathbf{x}(t) \in \mathcal{X}_\xi, \forall t \in [\tau_0, \tau_v)$. □

*4) Thrust Constraints $\mathbf{u} \in \mathcal{U}_T$:* Upper-bounding the total thrust is needed to prevent actuator saturation, while lower-bounding the total thrust is common in aerospace systems for soft-landing [47]. Consider the following thrust constraints set:

$$\mathcal{U}_T = \{\mathbf{u} \in \mathbb{R}^4 \mid \underline{T} \leq T \leq \overline{T}\}, \tag{26}$$

where $\underline{T}$, $\overline{T}$ are given constants satisfying $0 \leq \underline{T} \leq g \leq \overline{T}$. The following lemma provides sufficient conditions that are convex on the 2nd order VCPs $\mathbf{P}_i^{(2)}$, such that the thrust constraint $\mathbf{u}(t) \in \mathcal{U}_T$ is satisfied for $t \in [\tau_0, \tau_v)$.

**Lemma 5.** *Consider B-spline curve $\mathbf{r}(t)$ defined as in* (5)*. If*

$$\left\|\mathbf{P}_j^{(2)} + g\mathbf{z}_W\right\|_2 \leq \overline{T}, \quad j = 2, \ldots, N, \tag{27a}$$

$$\mathbf{z}_W^T \mathbf{P}_j^{(2)} \geq \underline{T} - g, \quad j = 2, \ldots, N, \tag{27b}$$

*hold, then $\mathbf{u}(t) \in \mathcal{U}_T, \ \forall t \in [\tau_0, \tau_v)$.*

*Proof.* By Proposition 1, and expanding the norm shown in (27a), we have:

$$\sqrt{\ddot{x}^2(t) + \ddot{y}^2(t) + \left(\ddot{z}(t) + g\right)^2} \leq \overline{T}, \quad t \in [\tau_0, \tau_v), \tag{28}$$

which constrains the acceleration to lie within a sphere in $\ddot{\mathbf{r}}$ coordinates centered at $(0, 0, -g)$. Note that by the flatness map (3), (28) is equivalent to $T(t) \leq \overline{T}, \ \forall t \in [\tau_0, \tau_v)$. Next, by Proposition 1, (27b) implies $\ddot{z}(t) \geq \underline{T} - g, \ \forall t \in [\tau_0, \tau_v)$, which lower-bounds the $z$-component of acceleration. Thus, $\underline{T} \leq \sqrt{\ddot{x}^2(t) + \ddot{y}^2(t) + \left(\ddot{z}(t) + g\right)^2} = T(t) \leq \overline{T}$ holds for $t \in [\tau_0, \tau_v)$. This completes the proof. □

**Remark 2.** *When the inequality $\underline{T} \leq T(t)$ is expressed in terms of the acceleration $\ddot{\mathbf{r}}(t)$ by* (3)*, it signifies exclusion from a spherical region. As a result, it is a nonconvex constraint in terms of the B-spline control points. Such a constraint was converted to a mixed-integer constraint in [12], and its conservative relaxation was also considered in [48]. The spherical region* (28) *is bigger than typical quadcopter operation requires: the lower half ($\ddot{z} < -g$) of sphere* (28) *is accessed only when the quadcopter is "upside-down" ($|\phi| \geq \pi/2$ or $|\theta| \geq \pi/2$) and this is only required in very aggressive flights. The proof of Lemma 5 reveals that some generally unused feasible region is sacrificed by lower-bounding $\ddot{z}(t)$ which results in a lower-bound for $T(t)$.*

*5) Angular Velocity Constraints $\mathbf{u} \in \mathcal{U}_\omega$:* Angular velocities are the inputs to the quadcopter model considered and are usually subject to saturation. Limits in angular velocities are also desirable to ensure integrity of gyroscope data. Consider the angular velocity constraints set defined as follows:

$$\mathcal{U}_\omega = \{\mathbf{u} \in \mathbb{R}^4 : |p|, |q| \leq \overline{\omega}\}, \tag{29}$$

where $\overline{\omega}$ is a given bound for the angular velocities. Recall that $r(t) = 0$ because of the assumption $\psi(t) = 0$. The following lemma provides sufficient conditions that are convex on the 2nd order VCPs $\mathbf{P}_i^{(2)}$ and 3rd order VCPs $\mathbf{P}_i^{(3)}$, such that the angular velocity constraint $\mathbf{u}(t) \in \mathcal{U}_\omega$ is satisfied for $t \in [\tau_0, \tau_v)$.

**Lemma 6.** *Consider B-spline curve $\mathbf{r}(t)$ defined as in* (5)*. For any vector $\boldsymbol{\zeta} = (\zeta_1, ..., \zeta_{N-d+1})^T$ whose entries are positive constants, if the following conditions:*

$$\mathbf{z}_W^T \mathbf{P}_j^{(2)} \geq \zeta_{l-d+1} - g, \quad j = l - d + 2, \ldots, l, \tag{30a}$$

$$\left\|\mathbf{P}_j^{(3)}\right\|_2 \leq \overline{\omega}\ \zeta_{l-d+1}, \quad j = l - d + 3, \ldots, l, \tag{30b}$$

*hold for $l = d, \ldots, N$, then $\mathbf{u}(t) \in \mathcal{U}_\omega, \ \forall t \in [\tau_0, \tau_v)$.*

*Proof.* By (16) from Proposition 1, we can establish

$$T(t) \geq \zeta_{l-d+1}, \ \|\dddot{\mathbf{r}}(t)\|_2 \leq \overline{\omega}\ \zeta_{l-d+1}, \ \forall t \in [\tau_l, \tau_{l+1}), \tag{31}$$

which lower-bounds the thrust and upper-bounds the magnitude of the jerk over the $l$-th knot sub-interval. Because the entries of $\boldsymbol{\zeta}$ are positive, we can combine inequalities to write $\|\dddot{\mathbf{r}}(t)\|_2 / T(t) \leq \|\dddot{\mathbf{r}}(t)\|_2 / \zeta_{l-d+1} \leq \overline{\omega}, \forall t \in [\tau_l, \tau_{l+1})$. Examining the flat maps for $p$ and $q$ given in (3) and taking the absolute value of both sides we have $|p| = |-\mathbf{y}_B \cdot \mathbf{h}_\omega|$ and $|q| = |\mathbf{x}_B \cdot \mathbf{h}_\omega|$. Since $\mathbf{x}_B$ and $\mathbf{y}_B$ are unit vectors, we can write $|p|, |q| \leq \|\mathbf{h}_\omega\|_2$ by the definition of the dot product. Furthermore, examining $\mathbf{h}_\omega$ we note that the numerator is a projection of the jerk to a plane perpendicular to the $\ddot{z}$-axis which only reduces its norm, and that the denominator is exactly the thrust $T$. In particular, $|p(t)|, |q(t)| \leq \|\mathbf{h}_\omega(t)\|_2 \leq \|\dddot{\mathbf{r}}(t)\|_2 / T(t) \leq \overline{\omega}$ holds for $t \in [\tau_l, \tau_{l+1})$. Taking $l = d, \ldots, N$ and letting $\boldsymbol{\tau}$ be a clamped knot vector satisfying (4a) leads to $|p(t)|, |q(t)| \leq \overline{\omega}, \forall t \in [\tau_0, \tau_v)$, which completes the proof. □

*6) Waypoint Constraints:* Waypoint constraints are commonly used for shaping the trajectory. Assume that there are $n_{wp}$ waypoints $\mathbf{p}_1^{wp}, \ldots, \mathbf{p}_{n_{wp}}^{wp} \in \mathbb{R}^3$ near which the trajectory must pass at certain discrete times $t_i, (i = 1, \ldots, n_{wp})$. Specifically, the following constrains are enforced for the position:

$$\|\mathbf{r}(t_i) - \mathbf{p}_i^{wp}\|_2 \leq d_{wp}, \quad i = 1, \ldots, n_{wp}, \tag{32}$$

where $d_{wp} \geq 0$ is the desired radius indicating the closeness of the trajectory to the waypoints. Note that the waypoint constraints are exact when $d_{wp} = 0$. Recalling that $\mathbf{r}(t)$ is considered a B-spline curve, constraint (32) is a SOC constraint in terms of its control points:

$$\left\|P\boldsymbol{\Lambda}_d(t_i) - \mathbf{p}_i^{wp}\right\|_2 \leq d_{wp}, \quad i = 1, \ldots, n_{wp}. \tag{33}$$

### B. SOCP-based Safe Trajectory Planning

Suppose that the initial and final states $\mathbf{x}_0, \mathbf{x}_f \in \mathbb{R}^9$ are given such that the trajectory must start at $\mathbf{x}_0$ and end at $\mathbf{x}_f$. This is accomplished with the following constraints on the VCPs:

$$\mathbf{P}_0 = \mathbf{r}_0, \quad \mathbf{P}_N = \mathbf{r}_f, \tag{34a}$$

$$\mathbf{P}_1^{(1)} = \dot{\mathbf{r}}_0, \quad \mathbf{P}_N^{(1)} = \dot{\mathbf{r}}_f, \tag{34b}$$

$$\mathbf{P}_2^{(2)} = g(\mathbf{z}_B^0 - \mathbf{z}_W), \quad \mathbf{P}_N^{(2)} = g(\mathbf{z}_B^f - \mathbf{z}_W), \tag{34c}$$

where, $\mathbf{z}_B^0 = (\cos\phi_0 \sin\theta_0, -\sin\phi_0, \cos\phi_0 \cos\theta_0)^T$, $\mathbf{z}_B^f = (\cos\phi_f \sin\theta_f, -\sin\phi_f, \cos\phi_f \cos\theta_f)^T$, $\mathbf{r}_0 = (x_0, y_0, z_0)^T$, $\mathbf{r}_f = (x_f, y_f, z_f)^T$.

We minimize $\int_{\tau_0}^{\tau_v} \|\mathbf{r}^{(4)}(t)\|_2^2 \mathrm{d}t$ where $\mathbf{r}^{(4)}(t)$ denotes the "snap" of the trajectory. By the flat map of the quadcopter, minimizing the snap is equivalent to minimizing the input effort [6]. Taking $\mathbf{r}(t)$ to be a B-spline curve constructed as in (5), the (simple) objective is given as

$$J_s(\mathbf{P}_0, \ldots, \mathbf{P}_N) = \int_{t_0}^{t_f} \left\| \sum_{i=0}^{N} \mathbf{P}_i \mathbf{b}_{d,4,i+1}^T \mathbf{\Lambda}_{d-4}(t) \right\|_2^2 \mathrm{d}t,$$

where the decision variables are the control points $\mathbf{P}_0, \ldots, \mathbf{P}_N$ of the B-spline curve $\mathbf{r}(t)$. Examining (31), we observe that it is desirable to make each entry $\zeta_k$ large to maximize the feasible region. Therefore, we introduce $\boldsymbol{\zeta}$ into the objective function such that it is maximized. Specifically, the modified objective function is chosen to be:

$$J(\mathbf{P}_0, \ldots, \mathbf{P}_N, \boldsymbol{\zeta}) = J_s(\mathbf{P}_0, \ldots, \mathbf{P}_N) - \sum_{k=1}^{N-d+1} \zeta_k. \tag{35}$$

Noting that the VCPs are linear combinations of the control points, we formulate the following SOCP to solve the safe trajectory planning problem:

$$\min_{\mathbf{P}_0, \ldots, \mathbf{P}_N, \boldsymbol{\zeta}} \quad J(\mathbf{P}_0, \ldots, \mathbf{P}_N, \boldsymbol{\zeta}) \qquad \text{(FLAT-SOCP)}$$

$$\text{s. t.} \quad (19), (21), (24), (27), (30), (33) \text{ and } (34) \text{ hold},$$

where "FLAT" means that the solutions parameterize the flat output trajectory $\boldsymbol{\sigma}(t) = (\mathbf{r}(t), 0)$.

The trajectory in the flat space is defined by control points generated from (FLAT-SOCP), and the corresponding state and input trajectories are obtained by passing the flat trajectory through the flat map (3). By Lemma 3, Lemma 4, Lemma 5 and Lemma 6, the resulting state and input trajectories of the quadcopter satisfy all the safety constraints in continuous time. This result is summarized in the following theorem.

**Theorem 1.** *Suppose that* (FLAT-SOCP) *has a feasible solution* $\mathbf{P}_0, \ldots, \mathbf{P}_N$. *Let* $\mathbf{r}(t)$ *be a B-spline curve defined by the control points* $\mathbf{P}_0, \ldots, \mathbf{P}_N$. *Then, the quadcopter state* $\mathbf{x}(t)$ *and input* $\mathbf{u}(t)$ *trajectories, which are obtained from the flat map* (3) *with* $\boldsymbol{\sigma}(t) = (\mathbf{r}(t), 0)$, *satisfy the safety constraints* (17) *for all* $t \in [\tau_0, \tau_v)$.

**Remark 3.** *Problem* (FLAT-SOCP) *can be relaxed to a QP if one replaces the convex regions described by the cone constraints in* (FLAT-SOCP) *with their polytopic inner approximations. For example, for the SOC constraint* (27a), *if we find a polytopic inner approximation as follows:* $\{\ddot{\mathbf{r}} \in \mathbb{R}^3 : A_T \ddot{\mathbf{r}} \leq \mathbf{b}_T\} \subset \{\ddot{\mathbf{r}} \in \mathbb{R}^3 :$ (28) *holds*$\}$, *then we can replace the SOC constraint* (27a) *with the linear constraint* $A_T \mathbf{P}_j^{(2)} \leq \mathbf{b}_T (j = 2, \ldots, N)$ *in* (FLAT-SOCP). *Similar replacements are possible for all SOC constraints.*

**Remark 4.** *The optimization problem* (FLAT-SOCP) *is based on B-spline parameterization of the flat trajectory, which is more convenient than the parameterization using piecewise polynomials as in [6]. By property P1), the smoothness requirement from the flatness map is satisfied automatically by choosing* $d \geq 4$, *without requiring additional smoothness constraints as in [6] or objective function reformulation as in [7]. Furthermore, the feasible region of* (FLAT-SOCP) *may be enlarged at the expense of computational burden by increasing* $N$ *(number of control points), which must respect* $v = N + d + 1$.

**Remark 5.** *Constraint* (30a) *is compatible with* (27a) *and* (27b). *When* $\underline{T}$ *is provided, it follows that* $\underline{T} \leq \zeta_k \leq \overline{T}, k = 1, \ldots, N - d + 1$. *In fact, the present formulation may lower-bound* $T(t)$ *more tightly than required to expand the feasible region for* (31), *but never so much that it violates the upper bound. Additionally, while constraint* (30) *increases optimality by expanding the feasible region, it comes at the expense of computational cost. A possible compromise is by replacing the vector* $\boldsymbol{\zeta}$ *with a scalar* $\zeta_\omega$ *in the objective function* (35) *and replacing constraint* (30) *with the global constraints:* $\mathbf{z}_W^T \mathbf{P}_j^{(2)} \geq \zeta_\omega - g (j = 2, \ldots, N)$ *and* $\left\|\mathbf{P}_j^{(3)}\right\|_2 \leq \overline{\omega}\ \zeta_\omega (j = 3, \ldots, N)$.

### C. Obstacle Avoidance in Continuous Time

In this subsection, we investigate how (FLAT-SOCP) can be modified to achieve obstacle avoidance with a rigorous continuous-time guarantee by leveraging the locality property of B-splines. The safe flight space is usually nonconvex, which makes the obstacle avoidance problem difficult [15]. Obstacle avoidance with continuous-time safety guarantees was studied in [14] where each obstacle is expressed as a polytope and the trajectory planning problem is formulated as a mixed-integer QP, which is an *NP-hard* problem and renders the online trajectory re-planning computationally infeasible.

Suppose that the safe flight space $\mathcal{F} \subset \mathbb{R}^3$ is a set defined as $\mathcal{F} \triangleq \mathbb{R}^3 \setminus (\mathcal{O}_1 \cup \ldots \cup \mathcal{O}_{n_o})$ where $\mathcal{O}_i \subset \mathbb{R}^3$ denotes an overapproximation of the $i$-th obstacle. Although $\mathcal{F}$ is generally nonconvex, mild assumptions about the distribution of the obstacles allow us to determine the existence of convex subsets that altogether form a *safe corridor* in $\mathcal{F}$ for the trajectory [49]. In practice, these convex subsets can be found using sensors such as stereo cameras and lidars [50]. In this paper, we assume that the overlapping convex sets of the safe corridor are known (e.g., using methods in [49]–[52]) and can be used for trajectory planning. The construction of these sets is considered the task of a high-level planner and is out of the scope of this work.

By utilizing the local property of B-spline curves, the following results provides sufficient conditions so that the trajectory remains in the safe corridor, which implies that the collision avoidance is achieved in continuous time.

**Proposition 2.** *Suppose that there exist $n_s$ overlapping convex sets $\{\mathcal{S}_1, \dots, \mathcal{S}_{n_s}\}$ inside the safe flight space, i.e., $\mathcal{S}_l \subset \mathcal{F},\ l = 1, \dots, n_s$, satisfying $\mathcal{S}_i \cap \mathcal{S}_{i+1} \neq \emptyset,\ i = 1, \dots, n_s - 1$ and consider B-spline curve $\mathbf{r}(t)$ defined as in (5) with $N = n_s + d - 1$. If the following conditions hold:*

$$\mathbf{P}_{j-1} \in \mathcal{S}_l, \quad j = l, \dots, l+d, \quad l = 1, \dots, n_s, \tag{36}$$

*then $\mathbf{r}(t) \in \mathcal{F}$ for all $t \in [\tau_0, \tau_v)$ and $\mathbf{r}(t)$ is at least $C^{d-1},\ \forall t \in [\tau_0, \tau_v)$.*

*Proof.* Applying Proposition 1 to the inclusion constraint of the $l$-th group of control points results in $\mathbf{r}(t) \in \mathcal{S}_l,\ t \in [\tau_{l+d-1}, \tau_{l+d})$, which means that the $l$-th segment of the B-spline curve $\mathbf{r}(t)$ is contained within $\mathcal{S}_l$. Taking now $l = 1, \dots, n_s$ and recalling that we consider a clamped knot vector (4a) results in $\mathbf{r}(t) \in \mathcal{F},\ t \in [\tau_0, \tau_v)$. Further, note that $\mathcal{S}_l \cap \mathcal{S}_{l+1} \neq \emptyset$ allows $\mathbf{r}(t)$ to be continuous despite the segment-wise constraints. □

The constraints shown in (36) can be readily added to (FLAT-SOCP) without compromising the problem structure if the convex sets $\{\mathcal{S}_l\}$ are restricted to SOC forms, as will be demonstrated in the experiments in Section VII.

**Remark 6.** *Convex sets $\mathcal{S}_1, \dots, \mathcal{S}_{n_s}$ in Proposition 2 play a significant role in the properties of the resulting B-spline curve (such as the magnitude of its derivatives). The optimal assignment of control points to each convex set remains an open problem and is still under our investigation. Relevant results include the time-optimality of trajectories generated using safe corridors [51], [52].*

**Remark 7.** *In Section VI we will establish safety guarantees for bounded reference position trajectory tracking in the infinity-norm sense. This bound at the tracking level can be incorporated when finding the sets $\{\mathcal{S}_l\}$ so that obstacle avoidance is guaranteed also during tracking. Alternatively, one can appropriately enlarge the size of obstacles $\{\mathcal{O}_i\}$.*

## VI. Trajectory Tracking with Continuous-Time Safety Guarantee

In this section, we propose a trajectory tracking method for quadcopters with position safety guarantees in continuous-time. The idea is to use CBFs as a safety filter to guarantee boundedness between the real trajectory and the nominal safe trajectory $\mathbf{r}^{\text{ref}}(t)$ generated from (FLAT-SOCP). The safe tracking controller is obtained by solving a convex QP online, whose feasibility will be ensured by the control-sharing property of multiple CBFs.

### A. Quadcopter Model for Safe Trajectory Tracking

We assume now that the angular velocity dynamics are regulated by some high-bandwidth controller and consider the reduced state $\mathbf{z} = (\mathbf{r}^T, \dot{\mathbf{r}}^T)^T$ and input $\mathbf{v} = (T, \boldsymbol{\xi}^T)^T$ subject to (1a). Then, we can consider a set of virtual inputs $\boldsymbol{\mu} = (\mu_1, \mu_2, \mu_3)^T \triangleq (\ddot{x}, \ddot{y}, \ddot{z})^T$ found as in (1a):

$$\boldsymbol{\mu} = \Psi(\mathbf{v}) \triangleq T\mathbf{z}_B - g\mathbf{z}_W. \tag{37}$$

Note that the map is invertible if we recover the yaw angle such that $\mathbf{v} = \Psi^{-1}(\boldsymbol{\mu}, \psi)$ where $\Psi^{-1}$ is given as:

$$T = \sqrt{\mu_1^2 + \mu_2^2 + (\mu_3 + g)^2} = \frac{\mu_3 + g}{\cos\phi\cos\theta}, \tag{38a}$$

$$\phi = \arctan\Big(\frac{\mu_1 \sin\psi - \mu_2 \cos\psi}{(\mu_3 + g)\cos\theta}\Big), \tag{38b}$$

$$\theta = \arctan\Big(\frac{\mu_1 \cos\psi + \mu_2 \sin\psi}{\mu_3 + g}\Big). \tag{38c}$$

With the virtual inputs, the quadcopter dynamics becomes a double-integrator [19], [27]:

$$\dot{\mathbf{z}}(t) = f(\mathbf{z}(t)) + g(\mathbf{z}(t))\boldsymbol{\mu}(t) = \underbrace{\begin{bmatrix} \mathbf{0}_3 & I_3 \\ \mathbf{0}_3 & \mathbf{0}_3 \end{bmatrix}}_{A} \mathbf{z}(t) + \underbrace{\begin{bmatrix} \mathbf{0}_3 \\ I_3 \end{bmatrix}}_{B} \boldsymbol{\mu}(t) \tag{39}$$

Assume that a reference state $\mathbf{x}^{\text{ref}}(t)$ and a reference input $\mathbf{u}^{\text{ref}}(t)$ are generated as in Sec. V and note that we can immediately extract $\mathbf{z}^{\text{ref}}(t)$ and $\mathbf{v}^{\text{ref}}(t)$ from them. The dynamic feasibility of the reference trajectory will be necessary to establish the forthcoming results. Assume also a *nominal controller* $\mathbf{v} = \pi(\mathbf{z})$ is given, which is potentially unsafe. In the following subsection, a CBF-QP-based tracking controller will be designed such that the real trajectory lies within a prescribed tube of the nominal trajectory while the tracking control is as close as possible to the nominal controller.

### B. CBF-QP-based Safe Tracking Controller

Consider the following time-varying safe set:

$$\mathcal{S}_h(t) = \{\mathbf{z} \in \mathbb{R}^6 : \|\mathbf{r} - \mathbf{r}^{\text{ref}}(t)\|_\infty \leq \delta\}, \quad t \in [\tau_0, \tau_v), \tag{40}$$

where $\delta > 0$ represents the maximum allowable deviation in each axis. That is, the trajectory is considered as safe if it is within a *safe tube* of radius $\delta$ around the nominal trajectory. Note that it is straightforward to generalize the following results to having different maximum bounds for each axis.

To ensure $\mathbf{z}(t) \in \mathcal{S}_h(t),\ \forall t \in [\tau_0, \tau_v)$, we define the following six candidate CBFs for $q \in \{x, y, z\}$:

$$\begin{bmatrix} h_{\overline{q}} \\ h_{\underline{q}} \end{bmatrix} \triangleq \begin{bmatrix} \delta + q^{\text{ref}}(t) - q(t) \\ \delta + q(t) - q^{\text{ref}}(t) \end{bmatrix}. \tag{41}$$

Note that all functions $h_{\overline{q}}, h_{\underline{q}},\ q \in \{x, y, z\}$ have relative degree 2. According to Definition 2, we choose constants $a_1, a_2$ such that the roots of $\lambda^2 + a_1\lambda + a_2$ are reals and negative, and the CBF conditions of relative degree 2 are satisfied for $h_{\overline{q}}, h_{\underline{q}}, \forall q \in \{x, y, z\}$. For example, the CBF

condition for $h_{\overline{x}}$ is $L_g\bar{L}_f h_{\overline{x}}\boldsymbol{\mu} + \bar{L}_f^2 h_{\overline{x}} + a_1\bar{L}_f h_{\overline{x}} + a_2 h_{\overline{x}} \geq 0$, which is equivalent to $\phi_{\overline{x}}^1\boldsymbol{\mu} + \phi_{\overline{x}}^2 \leq 0$, where $\phi_{\overline{x}}^1 = \mathbf{x}_W^T$ and $\phi_{\overline{x}}^2 = -\ddot{x}^{\text{ref}}(t) + a_1\big(\dot{x} - \dot{x}^{\text{ref}}(t)\big) + a_2\big(x - x^{\text{ref}}(t) - \delta\big)$.

Putting the six CBF conditions (for $h_{\overline{x}}, h_{\underline{x}}, h_{\overline{y}}, h_{\underline{y}}, h_{\overline{z}}, h_{\underline{z}}$), which are all linear constraints on $\boldsymbol{\mu}$ for fixed $\mathbf{z}$, into a QP, and choosing the objective function such that $\|\Psi(\pi(\mathbf{z})) - \boldsymbol{\mu}\|_2^2$ is minimized, where $\Psi(\pi(\mathbf{z}))$ yields the nominal virtual inputs, we have the following CBF-QP:

$$\begin{aligned} \min_{\boldsymbol{\mu}} \quad & \|\Psi\big(\pi(\mathbf{z})\big) - \boldsymbol{\mu}\|_2^2 \qquad \text{(CBF-QP)} \\ \text{s. t.} \quad & \phi_{\overline{q}}^1\boldsymbol{\mu} + \phi_{\overline{q}}^2 \leq 0, \\ & \phi_{\underline{q}}^1\boldsymbol{\mu} + \phi_{\underline{q}}^2 \leq 0, \quad q \in \{x, y, z\}, \end{aligned}$$

where the virtual input $\boldsymbol{\mu}$ is the decision variable, $\phi_{\overline{q}}^1 = \mathbf{q}_W^T$, $\phi_{\overline{q}}^2 = -\ddot{q}^{\text{ref}}(t) + a_1\big(\dot{q} - \dot{q}^{\text{ref}}(t)\big) + a_2\big(q - q^{\text{ref}}(t) - \delta\big)$, $\phi_{\underline{q}}^1 = -\mathbf{q}_W^T$ and $\phi_{\underline{q}}^2 = \ddot{q}^{\text{ref}}(t) + a_1\big(\dot{q}^{\text{ref}}(t) - \dot{q}\big) + a_2\big(q^{\text{ref}}(t) - q - \delta\big)$, with $\mathbf{q}_W$ representing the appropriate world-frame's axis (see Figure 2). Note that by the dynamic feasibility of the reference trajectory, we can extract $\ddot{q}^{\text{ref}}(t)$ from $\dot{\mathbf{z}}^{\text{ref}}(t) = A\mathbf{z}^{\text{ref}}(t) + B\Psi\big(\mathbf{v}^{\text{ref}}(t)\big)$ for any $t \in [\tau_0, \tau_v)$.

The (CBF-QP) is convex and can be solved in polynomial-time by QP solvers such as MOSEK [45] or OSQP [53]. Letting $\boldsymbol{\mu}^*(t)$ be the solution of (CBF-QP) for any $(t, \mathbf{z}) \in [\tau_0, \tau_v) \times \mathbb{R}^6$, the safe input to the quadcopter is then given as

$$\mathbf{v}_s(t) = \Psi^{-1}\big(\boldsymbol{\mu}^*(t), \psi(t)\big), \tag{42}$$

where $\psi(t)$ is recovered from $\mathbf{v}(t) = \pi\big(\mathbf{z}(t)\big)$. The following theorem provides sufficient conditions that ensure the forward invariance of the set $\mathcal{S}_h(t)$, which implies the assurance of the safe trajectory tracking, as well as the feasibility of (CBF-QP).

**Theorem 2.** *Consider the system given in* (39)*. If constants $a_1, a_2 > 0$ are chosen such that the roots of $\lambda^2 + a_1\lambda + a_2$ are negative reals $-\lambda_1, -\lambda_2 < 0$, then* (CBF-QP) *is feasible for all $(t, \mathbf{z}) \in [\tau_0, \tau_v) \times \mathbb{R}^6$. Furthermore, if $\|\mathbf{r}(\tau_0) - \mathbf{r}^{ref}(\tau_0)\|_\infty \leq \delta$, $\|\dot{\mathbf{r}}(\tau_0) - \dot{\mathbf{r}}^{ref}(\tau_0) + \lambda_1(\mathbf{r}(\tau_0) - \mathbf{r}^{ref}(\tau_0))\|_\infty \leq \lambda_1\delta$, and $\boldsymbol{\mu}^*(t)$ from* (CBF-QP) *is locally Lipchitz in $\mathbf{z}$, then the input $\mathbf{v}_s(t)$ will render $\|\mathbf{r}(t) - \mathbf{r}^{ref}(t)\|_\infty \leq \delta$ for all $t \in [\tau_0, \tau_v)$. Therefore, $\mathbf{z}(t) \in \mathcal{S}_h(t)$ for all $t \in [\tau_0, \tau_v)$.*

*Proof.* The candidate CBFs defined in (41) aim to ensure $q^{\text{ref}}(t) - \delta \triangleq \underline{q}(t) \leq q(t) \leq \overline{q}(t) \triangleq q^{\text{ref}} + \delta$, for $q \in \{x, y, z\}$. Recalling that $\boldsymbol{\mu} = [\mu_1, \mu_2, \mu_3]^T$, the system model shown in (39) is equivalent to $\ddot{x} = \mu_1$, $\ddot{y} = \mu_2$, $\ddot{z} = \mu_3$. Thus, the axes are decoupled which enable us to consider each axis individually. It is easy to see that for $q \in \{x, y, z\}$, $\overline{q} - \underline{q} = 2\delta$, $\dot{\overline{q}} - \dot{\underline{q}} = 0$, and $\ddot{\overline{q}} - \ddot{\underline{q}} = 0$. Since $a_1, a_2 > 0$, we have $\ddot{\overline{q}} - \ddot{\underline{q}} + a_1(\dot{\overline{q}} - \dot{\underline{q}}) + a_2(\overline{q} - \underline{q}) = 2a_2\delta > 0$. Therefore, condition (15) in Theorem 2 of [42] is satisfied, which implies that $h_{\overline{q}}, h_{\underline{q}}$ are control sharing barrier functions. It is easy to see that functions $h_{\overline{x}}, h_{\underline{x}}, h_{\overline{y}}, h_{\underline{y}}, h_{\overline{z}}, h_{\underline{z}}$ are all control sharing barrier functions, which means that (CBF-QP) is feasible for all $(t, \mathbf{z}) \in [\tau_0, \tau_v) \times \mathbb{R}^6$. Since $\|\mathbf{r}(\tau_0) - \mathbf{r}^{\text{ref}}(\tau_0)\|_\infty \leq \delta$, $\|\dot{\mathbf{r}}(\tau_0) - \dot{\mathbf{r}}^{\text{ref}}(\tau_0) + \lambda_1\big(\mathbf{r}(\tau_0) - \mathbf{r}^{\text{ref}}(\tau_0)\big)\|_\infty \leq \lambda_1\delta$, we have $|q(\tau_0) - q^{\text{ref}}(\tau_0)| \leq \delta$, $|\dot{q}(\tau_0) - \dot{q}^{\text{ref}}(\tau_0) + \lambda_1 q(\tau_0) - \lambda_1 q^{\text{ref}}(\tau_0)| \leq \lambda_1\delta$, $\forall q \in \{x, y, z\}$, which is equivalent to $h_{\overline{q}}(\tau_0) \geq 0$, $h_{\underline{q}}(\tau_0) \geq 0$ and $\dot{h}_{\overline{q}}(\tau_0) + \lambda_1 h_{\overline{q}}(\tau_0) \geq 0$, $\dot{h}_{\underline{q}}(\tau_0) + \lambda_1 h_{\underline{q}}(\tau_0) \geq 0$ for $q \in \{x, y, z\}$. Therefore, $h_{\overline{q}}(t) \geq 0$, $h_{\underline{q}}(t) \geq 0$ for all $t \in [\tau_0, \tau_v)$ and $q \in \{x, y, z\}$ by [42], which implies that $\|\mathbf{r}(t) - \mathbf{r}^{\text{ref}}(t)\|_\infty \leq \delta$ holds for $t \in [\tau_0, \tau_v)$. The final conclusion follows by the definition of set $\mathcal{S}_h(t)$. □

**Remark 8.** *The safety guarantee provided by Theorem 2 is predicated on the assumption that the control input generated by the* (CBF-QP) *can be updated continuously. When a sampled-data controller is implemented in practice, it is still possible to provide the same safety guarantee in continuous time; please refer to [54] which proposed a real-time implementable, robust sampled-data CBF-QP-based controller.*

For the trajectory obtained from (CBF-QP), we can also quantify the upper bounds of $\|\dot{\mathbf{r}}(t) - \dot{\mathbf{r}}^{\text{ref}}(t)\|_\infty$ and $\|\boldsymbol{\mu} - \ddot{\mathbf{r}}^{\text{ref}}(t)\|_\infty$ as shown in the following two corollaries.

**Corollary 1.** *If the conditions of Theorem 2 are satisfied and $\|\dot{\mathbf{r}}(\tau_0) - \dot{\mathbf{r}}^{ref}(\tau_0)\|_\infty \leq 2\delta a_2/a_1$, then $\|\dot{\mathbf{r}}(t) - \dot{\mathbf{r}}^{ref}(t)\|_\infty \leq 2\delta a_2/a_1$ holds for any $t \in [\tau_0, \tau_v)$.*

*Proof.* Any feasible solution $\boldsymbol{\mu} = (\dddot{x}, \dddot{y}, \dddot{z})^T$ of (CBF-QP) satisfies $\ddot{q} \leq \ddot{q}^{\text{ref}} + a_1(\dot{q}^{\text{ref}} - \dot{q}) + a_2(q^{\text{ref}} - q + \delta)$ and $\ddot{q} \geq \ddot{q}^{\text{ref}} + a_1(\dot{q}^{\text{ref}} - \dot{q}) + a_2(q^{\text{ref}} - q - \delta)$ for $q \in \{x, y, z\}$. Define candidate CBFs $\hat{h}_{\overline{q}}(t) \triangleq \dot{q}^{\text{ref}}(t) - \dot{q}(t) + 2\delta a_2/a_1$ and $\hat{h}_{\underline{q}}(t) \triangleq \dot{q}(t) - \dot{q}^{\text{ref}}(t) + 2\delta a_2/a_1$. Since $|q - q^{\text{ref}}| \leq \delta$, we have $\ddot{q} \leq \ddot{q}^{\text{ref}} + a_1(\dot{q}^{\text{ref}} - \dot{q}) + 2a_2\delta$, $\ddot{q} \geq \ddot{q}^{\text{ref}} + a_1(\dot{q}^{\text{ref}} - \dot{q}) - 2a_2\delta$. These two inequalities are equivalent to $\dot{\hat{h}}_{\overline{q}}(t) + a_1\hat{h}_{\overline{q}}(t) \geq 0$ and $\dot{\hat{h}}_{\underline{q}}(t) + a_1\hat{h}_{\underline{q}}(t) \geq 0$, respectively for $q \in \{x, y, z\}$ and any $t \in [\tau_0, \tau_v)$. Therefore, $\hat{h}_{\overline{q}}$ and $\hat{h}_{\underline{q}}$ satisfy the CBF condition. Furthermore, since $\|\dot{\mathbf{r}}(\tau_0) - \dot{\mathbf{r}}^{\text{ref}}(\tau_0)\|_\infty \leq 2\delta a_2/a_1$, we have $\hat{h}_{\overline{q}}(\tau_0) \geq 0$ and $\hat{h}_{\underline{q}}(\tau_0) \geq 0$ for $q \in \{x, y, z\}$. Thus, $\hat{h}_{\overline{q}}(t) \geq 0$ and $\hat{h}_{\underline{q}}(t) \geq 0$ for $t \in [\tau_0, \tau_v)$ by [23], which implies that $|\dot{q}^{\text{ref}}(t) - \dot{q}(t)| \leq 2\delta a_2/a_1$ for $q \in \{x, y, z\}$ and $t \in [\tau_0, \tau_v)$. The conclusion follows immediately. □

**Corollary 2.** *If the conditions of Theorem 2 are satisfied, then the feasible solution $\boldsymbol{\mu}$ of* (CBF-QP) *is bounded by $\|\boldsymbol{\mu} - \ddot{\mathbf{r}}^{ref}(t)\|_\infty \leq 4\delta a_2$ for any $t \in [\tau_0, \tau_v)$.*

*Proof.* Recall that $\|\mathbf{r}(t) - \mathbf{r}^{\text{ref}}(t)\|_\infty \leq \delta$ by Theorem 2 and $\|\dot{\mathbf{r}}(t) - \dot{\mathbf{r}}^{\text{ref}}(t)\|_\infty \leq 2\delta a_2/a_1$ by Corollary 1. These bounds allow us to bound the feasible set of (CBF-QP) by $\ddot{q}^{\text{ref}}(t) - 4\delta a_2 \leq \ddot{q} \leq \ddot{q}^{\text{ref}}(t) + 4\delta a_2$, where $q \in \{x, y, z\}$. Recalling that $\boldsymbol{\mu} = (\dddot{x}, \dddot{y}, \dddot{z})^T$ concludes the proof. □

### C. Input Constraints Satisfaction for the CBF-QP Controller

The thrust $T$, the roll angle $\phi$, and the pitch angle $\theta$ are related to the decision variable $\boldsymbol{\mu}$ in (CBF-QP) through the relation shown in (42). In this subsection, we will consider how the safety constraints on $T, \phi, \theta$ considered in Section V can be respected by the CBF-QP-based tracking controller that filters the nominal controller $\pi(\mathbf{z})$. Specifically, we aim to guarantee that $\mathbf{v}_s(t) \in \mathcal{V}$ where $\mathbf{v}_s(t)$ is given in (42) and

$$\mathcal{V} \triangleq \mathcal{V}_T \cap \mathcal{V}_\xi, \tag{43}$$

where $\mathcal{V}_T = \{\mathbf{v} \in \mathbb{R}^4 \mid 0 \leq T \leq \overline{T}\}$ and $\mathcal{V}_\xi = \{\mathbf{v} \in \mathbb{R}^4 : |\phi|, |\theta| \leq \epsilon < \pi/2\}$. Note that the constraint $\mathbf{v}_s(t) \in \mathcal{V}$ is important to ensure actuator saturation will not happen.

The following proposition presents sufficient conditions on the trajectory such that the safe input $\mathbf{v}_s(t)$ given in (42) respects the constraints shown in (43).

**Proposition 3.** *Consider the system given in* (39) *and suppose that the conditions of Theorem 2 are satisfied. If trajectories of the state* $\mathbf{z}^{ref}(t)$ *and the input* $\mathbf{v}^{ref}(t)$ *satisfy* (1a) *and the following conditions hold for all* $t \in [\tau_0, \tau_v)$*:*

$$0 \leq T^{ref}(t) \leq \overline{T} - 4\sqrt{3}\delta a_2, \tag{44a}$$

$$\|A_\epsilon \ddot{\mathbf{r}}^{ref}(t)\|_2 \leq \ddot{z}^{ref}(t) + g - 4\delta a_2(1 + |\cot\epsilon|\sqrt{2}), \tag{44b}$$

*with* $A_\epsilon = diag(\cot\epsilon, \cot\epsilon, 0)$*, then* $\mathbf{v}_s(t) \in \mathcal{V}$ *with* $\mathbf{v}_s(t)$ *given in* (42) *and* $\mathcal{V}$ *given in* (43)*.*

*Proof.* Let $\boldsymbol{\mu}^*(t)$ be the solution of (CBF-QP). By norm equivalence, $\|\boldsymbol{\mu} - \ddot{\mathbf{r}}^{\text{ref}}(t)\|_\infty \leq 4\delta a_2$ in Corollary 2 implies $\|\boldsymbol{\mu}^*(t) - \ddot{\mathbf{r}}^{\text{ref}}(t)\|_2 \leq \sqrt{3}\|\boldsymbol{\mu}^*(t) - \ddot{\mathbf{r}}^{\text{ref}}(t)\|_\infty \leq 4\sqrt{3}\delta a_2$ for any $t \in [\tau_0, \tau_v)$.

Now consider the thrust element of $\mathbf{v}_s(t)$ given as $T(t) = \|\boldsymbol{\mu}^*(t) + g\mathbf{z}_W\|_2$ by the map $\Psi^{-1}$. By (44a) we have $T^{\text{ref}}(t) + 4\sqrt{3}\delta a_2 = \|\ddot{\mathbf{r}}^{\text{ref}}(t) + g\mathbf{z}_W\|_2 + 4\sqrt{3}\delta a_2 \leq \overline{T}$. Therefore, $T(t) - \overline{T} = \|\boldsymbol{\mu}^*(t) + g\mathbf{z}_W\|_2 - \overline{T} \leq \|\boldsymbol{\mu}^*(t) + g\mathbf{z}_W\|_2 - \|\ddot{\mathbf{r}}^{\text{ref}}(t) + g\mathbf{z}_W\|_2 - 4\sqrt{3}\delta a_2 \leq \|\boldsymbol{\mu}^*(t) - \ddot{\mathbf{r}}^{\text{ref}}(t)\|_2 - 4\sqrt{3}\delta a_2 \leq 0$, Thus, $T(t) \leq \overline{T}$ for $t \in [\tau_0, \tau_v)$. Note that $\ddot{\mathbf{r}}^{\text{ref}}(t)$ can be extracted from $\dot{\mathbf{z}}^{\text{ref}}(t)$ which is given by dynamic feasibility of the reference trajectory.

Next, we examine the roll $\phi(t)$ and pitch $\theta(t)$ elements of $\mathbf{v}_s(t)$. We drop the time-dependence of terms for convenience. Let $\mathcal{K}_\epsilon$ be given as in (23). Note that

$$\begin{aligned}\left\|\begin{bmatrix}\mu_1^*\\ \mu_2^*\end{bmatrix}\right\|_2 &\leq \left\|\begin{bmatrix}\mu_1^*\\ \mu_2^*\end{bmatrix} - \begin{bmatrix}\ddot{x}^{\text{ref}}\\ \ddot{y}^{\text{ref}}\end{bmatrix}\right\|_2 + \left\|\begin{bmatrix}\ddot{x}^{\text{ref}}\\ \ddot{y}^{\text{ref}}\end{bmatrix}\right\|_2 \\ &\leq 4\sqrt{2}\delta a_2 + \left\|\begin{bmatrix}\ddot{x}^{\text{ref}}\\ \ddot{y}^{\text{ref}}\end{bmatrix}\right\|_2,\end{aligned} \tag{45}$$

where the first inequality is from the triangle inequality and the second inequality is from norm equivalence and Corollary 2. Therefore, $\|A_\epsilon\boldsymbol{\mu}^*\|_2 = |\cot\epsilon|\left\|(\mu_1^*, \mu_2^*)^T\right\|_2 \leq 4|\cot\epsilon|\sqrt{2}\delta a_2 + \|A_\epsilon\ddot{\mathbf{r}}^{\text{ref}}\|_2 \leq \ddot{z}^{\text{ref}} - 4\delta a_2 + g \leq \mu_3^* + g$ where the first inequality is from (45) and the definition of $A_\epsilon$, the second inequality is from (44b), and the third inequality is from the fact that $\ddot{z}^{\text{ref}} \leq \mu_3^* + 4\delta a_2$, which can be derived from $|\mu_3^* - \ddot{z}^{\text{ref}}| \leq 4\delta a_2$ by Corollary 2. Thus, $\mu^*(t) \in \mathcal{K}_\epsilon$ and, by Lemma 3, we have that $|\phi(t)|, |\theta(t)| \leq \epsilon$ for all $t \in [\tau_0, \tau_v)$. This completes the proof. □

**Remark 9.** *The conditions* (44a)-(44b) *for the reference trajectory can be easily guaranteed in continuous-time with the method presented in Section V, and* (FLAT-SOCP) *remains convex. Intuitively, conditions* (44a)-(44b) *shrink the cone* (23) *and the sphere* (28) *constraints on the trajectory acceleration (see Figure 3) enough so that all feasible solutions of* (CBF-QP) *satisfy the desired safety constraints when transformed through the map* $\Psi^{-1}$*. This means that, for any* $\psi$ *and any* $t \in [\tau_0, \tau_v)$*, the transformed optimal solution* $\mathbf{v}_s(t) = \Psi^{-1}(\boldsymbol{\mu}^*(t), \psi) \in \mathcal{V}$*.*

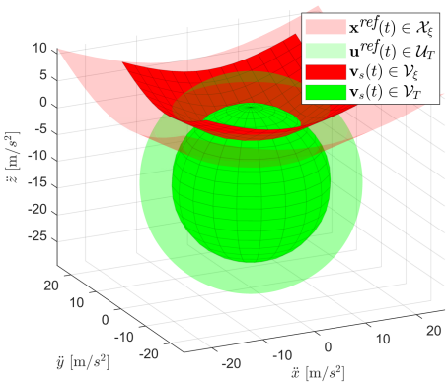



Fig. 3. Conditions (44a)-(44b) shrink the cone that bounds the Euler angles ($\mathbf{x}^{\text{ref}}(t) \in \mathcal{X}_\xi$) and the sphere that bounds the thrust ($\mathbf{u}^{\text{ref}}(t) \in \mathcal{U}_T$) to guarantee that safe inputs $\mathbf{v}_s(t) \in \mathcal{V}$. This figure shows the shrinkage with typical bounds $\epsilon = 1.0472$, $\overline{T} = 2g$ and the same safety parameters $\delta = 0.1$, $a_2 = 8$ that we use in Examples 1, 3 and 4 in Sec. VII.

## VII. Simulations & Experiments

In this section, we use simulation and experimental examples to demonstrate the effectiveness and versatility of the proposed approach. For all experiments, (FLAT-SOCP) is solved in MATLAB with Yalmip [55] and solver MOSEK [45] at a rate of 30-100 Hz depending on its complexity, and (CBF-QP) is solved in Python3 with OSQP [53] at a rate of 100 Hz. The tracking data was obtained from an OptiTrack motion capture system and filtered to obtain the state estimation by using an Extended Kalman Filter. All online computations are performed offboard in a basestation computer running Ubuntu 20.04 and ROS Noetic. The basestation is equipped with an Intel i9 @ 3.70 GHz CPU and 16 Gb RAM. The control inputs are sent to the Crazyflie2.1 over Crazyradio at an effective rate of 100 Hz, while the onboard attitude control and estimation are performed by Crazyflie2.1's firmware. The video of the experiments can be found in [56].

**Example 1.** *This example demonstrates rigorous satisfaction of the safety constraints in continuous-time by the SOCP-based reference trajectory as well as the CBF-QP-based tracking controller. First, we solve* (FLAT-SOCP) *to generate a reference trajectory satisfying* (17) *where the following safety constraints are required to be respected for all* $t \in [\tau_0, \tau_v)$*:*

$$\begin{aligned}&\|\dot{\mathbf{r}}^{ref}(t)\| \leq \overline{v}, \qquad |\phi^{ref}(t)|, |\theta^{ref}(t)| \leq \epsilon,\\ &\underline{T} \leq T^{ref}(t) \leq \overline{T}, \qquad |p^{ref}(t)|, |q^{ref}(t)| \leq \overline{\omega}.\end{aligned}$$

*The parameters of the trajectory and the safety constraints are given in Table II, and the parameters of the waypoints are given in Table III. With these parameters, we solve* (FLAT-SOCP) *and show the resulting position trajectory* $\mathbf{r}^{ref}(t)$ *in Figure 4 along with the waypoints and control points. The geometry of the constraints enforced for the* 2*nd order*

TABLE II
TRAJECTORY PARAMETERS FOR EXAMPLE 1

| | | | | | |
|---|---|---|---|---|---|
| $\tau_0$ [s] | 0 | $\mathbf{x}_0, \mathbf{x}_f$ [-] | $\mathbf{0}_{9\times1}$ | $\epsilon$ [deg] | 1.75 |
| $\tau_v$ [s] | 30 | $n_{wp}$ | 8 | $\underline{T}$ $[m/s^2]$ | 9.7 |
| N [-] | 40 | $d_{wp}$ [m] | 0.05 | $\overline{T}$ $[m/s^2]$ | 9.9 |
| d [-] | 5 | $\overline{v}$ $[m/s]$ | 0.5 | $\overline{\omega}$ [deg/s] | 1.5 |

TABLE III
TRAJECTORY WAYPOINTS FOR EXAMPLE 1

| $i$ | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|---|---|---|---|---|---|---|---|---|
| $\mathbf{p}_i^{wp}$ | -0.15 | -0.75 | 0.65 | 0.65 | -0.5 | -0.6 | 0.4 | 0.25 |
| | 0.25 | 0.6 | -0.65 | 0.5 | 0.5 | -0.6 | -0.4 | 0.25 |
| | 0.25 | 0.5 | 0.25 | 0.25 | 0.75 | 0.5 | 0.4 | 0.25 |
| $t_i$ | 4.5 | 7.8 | 12.6 | 15.3 | 18 | 21 | 24 | 27 |

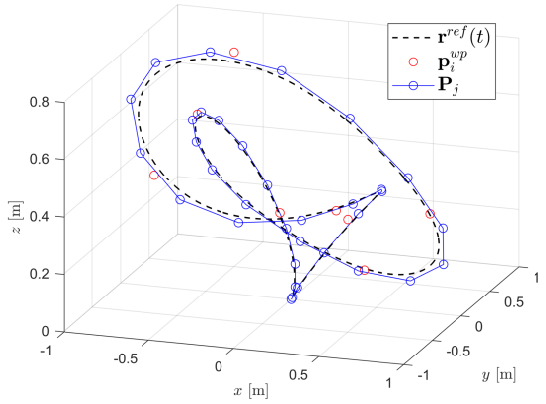



Fig. 4. The position trajectory generated in Example 1 with the parameters in Tables II and III. We show the position trajectory (dashed black) along with the waypoints (red circles) and the control points (connected, blue circles).

*VCPs $\mathbf{P}_j^{(2)}$ is shown in Figure 5, which can be interpreted as follows: (i) Inclusion in the conic surface (red) given by $\mathcal{K}_\epsilon$ defined in* (23) *to guarantee $\mathbf{x}^{ref}(t) \in \mathcal{X}_\xi$; (ii) Inclusion in the sphere (green) given by* (27a) *to guarantee upper-bounded thrust $T(t) \leq \overline{T}$; (iii) Restriction above hyperplane (cyan) given by* (27b) *to guarantee lower-bounded thrust $\underline{T} \leq T(t)$. The safety constraint satisfaction is verified in Figure 6 by transforming the flat output trajectory $\boldsymbol{\sigma}^{ref}(t) = (\mathbf{r}^{ref}(t), 0)$ via* (3) *to the state $\mathbf{x}^{ref}(t)$ and input $\mathbf{u}^{ref}(t)$ trajectories. From Figure 6, it can be observed that all considered safety constraints are satisfied in continuous-time. It can be also seen that the values of the vector $\boldsymbol{\zeta} = (\zeta_1, ..., \zeta_{36})^T$, which is illustrated by the dashed, light red line, lower-bound the thrust over local segments as defined in* (30).

*Next, we show that, when filtered through the* (CBF-QP), *a nominal controller $\pi(\mathbf{z})$ that is potentially unsafe can achieve safe trajectory tracking with $\delta = 0.1$ defined in* (40). *The nominal tracking controller $\pi$ can be arbitrarily chosen, but in this example we choose $\pi$ as an LQR controller with a large input weight to minimize its actuation effort (i.e., power consumption). We choose parameters $a_1 = 6, a_2 = 8$ and compare the performance of the CBF-QP-based safe tracking*

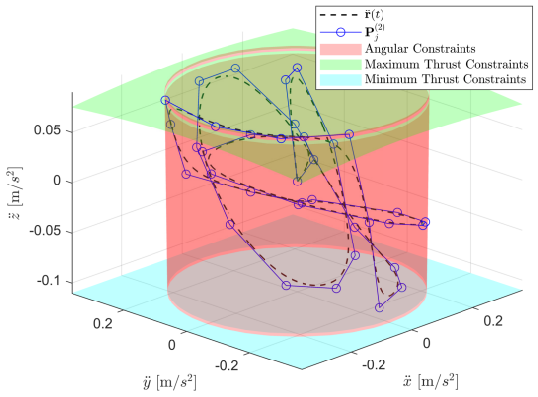
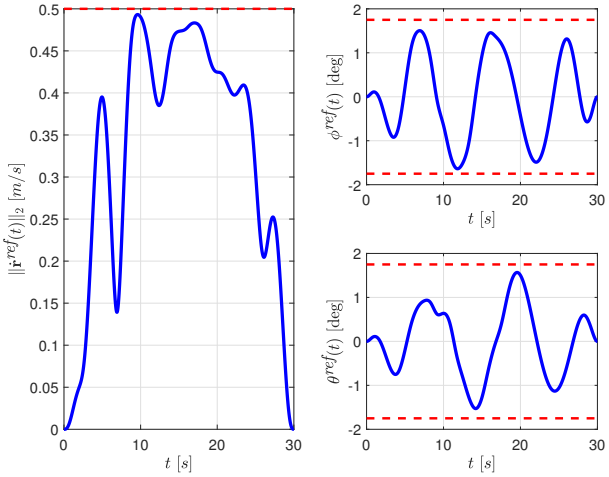
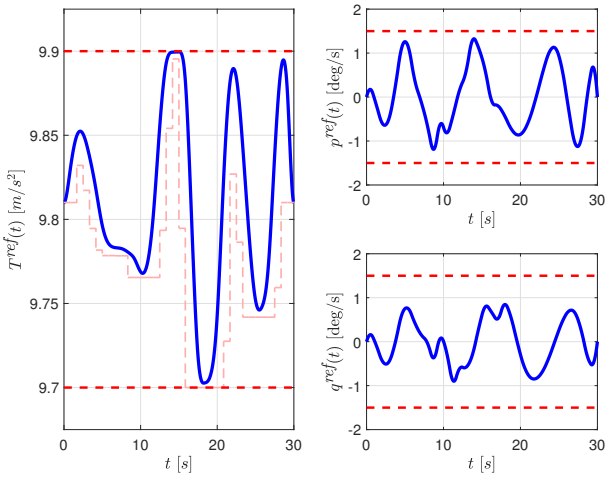



Fig. 5. Visualization of constraints enforced for the 2nd order VCPs $\mathbf{P}_j^{(2)}$ in Example 1. Constraint (24) represents inclusion within the red cone. Constraint (27a) is the inclusion within the green sphere (only cusp is shown). Constraint (27b) forces the VCPs to lie above the cyan plane (global bound). Note that the local counterpart (30a) is not depicted here. Note also that these are not the only constraints added to (FLAT-SOCP) in Example 1.




(a) State constraints. The velocity respects $\|\dot{\mathbf{r}}(t)\| \leq 0.5$ m/s and the Euler angles respect $|\phi(t)|, |\theta(t)| \leq 1.75$ deg for all $t \in [0, 30)$.

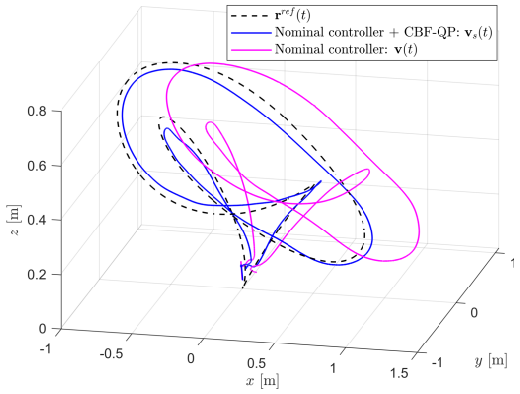



(b) Input constraints. The thrust respects $9.7$ m/$s^2 \leq T(t) \leq 9.9$ m/$s^2$ and the angular velocities respect $|p(t)|, |q(t)| \leq 1.5$ deg/s. The light red line indicates the values of the vector $\boldsymbol{\zeta} = (\zeta_1, \ldots, \zeta_{36})^T$ lower-bounding the thrust over local segments as shown in (30).

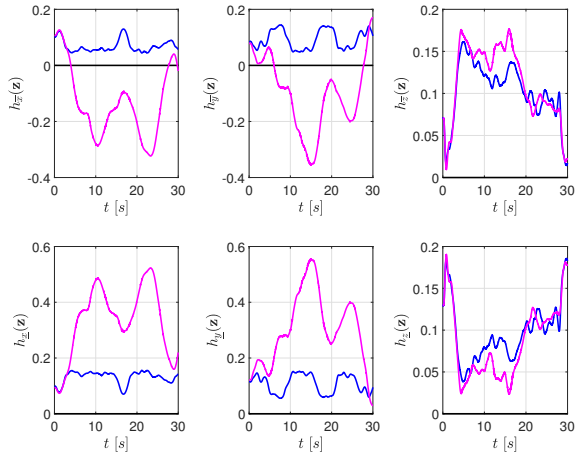
Fig. 6. Verification that the safety constraints for state and inputs in Example 1 are satisfied in continuous-time.

*controller and the unfiltered nominal controller in Figure 7. As can be seen from the figure, the safe bounding constraints are satisfied by the CBF-QP-based controller as CBFs defined in* (41) *satisfy* $h_{\overline{q}}, h_{\underline{q}} \geq 0$ *for* $q \in \{x, y, z\}$ *for all time, while these constraints are not respected by the nominal controller.*

**Example 2.** *In this example, we compare our SOCP-based trajectory generation approach against three other existing approaches: the flatness-based approach using piecewise polynomials given in [6], the open-source optimal control solver OpenOCL [30], and the approach based on motion primitives [31]. We compare the four approaches in terms of solve time and satisfaction of the safety constraints. For each approach, we aim to generate a trajectory with the following parameters and safety bounds, enforced as applicable: initial* $t_0 = 0$ *and final* $t_f = 10$ *times; initial and final states* $\mathbf{x}_0 = \mathbf{x}_f = \mathbf{0}_{9\times 1}$*; waypoint distance* $d_{wp} = 0.05$*; maximum speed* $\overline{v} = 1$*; angular bound* $\epsilon = 7$ *degrees; thrust bounds* $\underline{T} = 0$ *and* $\overline{T} = 10.5$*;*




(a) Tracking performance of the nominal LQR controller $\pi(\mathbf{z})$ with (blue) and without (magenta) the (CBF-QP).




(b) Measured values of functions $h_{\overline{x}}, h_{\underline{x}}, h_{\overline{y}}, h_{\underline{y}}, h_{\overline{z}}, h_{\underline{z}}$ defined in (41) when tracking with the filtered (blue) and unfiltered (magenta) nominal controller $\pi(\mathbf{z})$. The CBF-QP-based tracking controller respects safety as the values of these functions are always non-negative, which means that the real trajectory is within the prescribed tube ($\delta = 0.1$) around the nominal trajectory. The nominal LQR tracking controller (magenta) violates safety as $h_{\overline{x}}(t, \mathbf{z}(t)) < 0$ and $h_{\overline{y}}(t, \mathbf{z}(t)) < 0$ for some $t$.

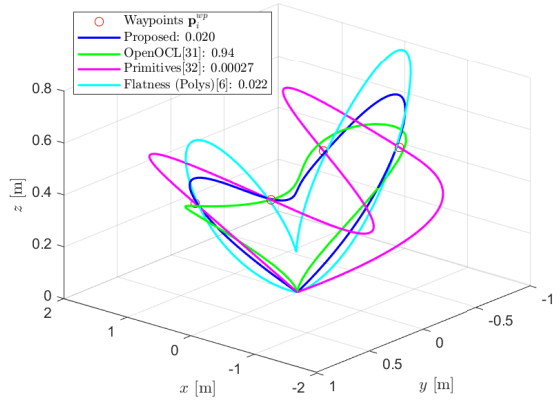
Fig. 7. Performance comparison of the nominal tracking controller and the CBF-QP-based safe tracking controller for Example 1.

*and maximum angular speed* $\overline{\omega} = 30$ *degrees per second. We also consider* $n_{wp} = 4$ *waypoints:* $\mathbf{p}_1^{wp} = (0.6, 0.6, 0.4)^T$, $\mathbf{p}_2^{wp} = (-0.6, 0.6, 0.5)^T$, $\mathbf{p}_3^{wp} = (0.6, -0.6, 0.4)^T$ *and* $\mathbf{p}_4^{wp} = (-0.6, -0.6, 0.5)^T$*. All computations are done in MATLAB except for the primitives [31] approach which is solved in Python3. The computer for this example was a laptop PC running Windows 10 with an Intel i5 @ 1.60 GHz CPU and 8 Gb RAM.*

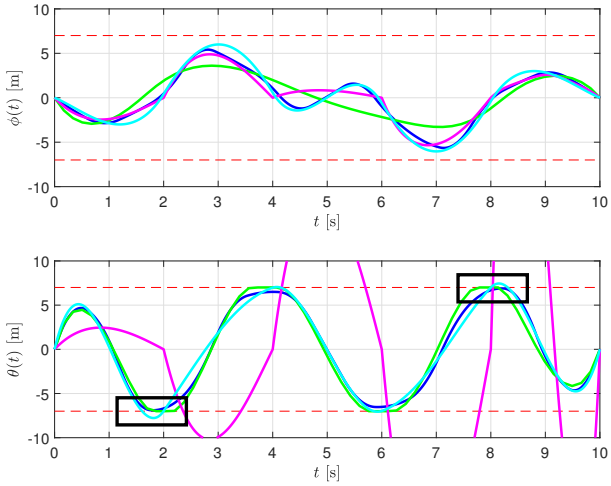
*The specific setups for the four approaches are given as follows. (i) For our SOCP-based approach, we choose* $d = 5$ *and* $N = 25$*. (ii) The flatness-based approach using piecewise polynomials in [6] involves solving a QP in terms of the polynomial coefficients. We augment this problem into a SOCP to include all safety constraints, but we only enforce these constraints at 25 discrete time instances evenly distributed in the interval* $[\tau_0, \tau_v]$*. The problem formulated remains convex because all presented convex conditions with respect to the (virtual) control points can be equivalently formulated with respect to the polynomial coefficients. In addition, we consider piecewise polynomials of degree 5 with 5 pieces. (iii) For the OpenOCL-based approach [30], formulating an optimal control problem for constrained trajectory generation is straightforward. We choose parameter* $N = 59$ *and objective function* $J = \int_{\tau_0}^{\tau_v} \left(T(t) - g\right)^2 + \|\boldsymbol{\omega}\|_2^2 \, \mathrm{d}t$*, and impose the state and input constraints shown in* (17) *directly except for* $\mathcal{X}_v$ *as OpenOCL was unable to handle this constraint. (iv) For the motion primitives approach in [31], it focuses on speed and does not admit any safety constraints at solve time. The waypoint constraints are handled by generating multiple trajectories and setting the start/end positions of each trajectory at the relevant waypoints.*

*The resulting trajectories for each approach are shown in Figure 8. The primitive-based approach is the fastest but induces large safety violations. The flatness-based approach using piecewise polynomials in [6] has similar performance as our method but fails to provide continuous-time safety guarantees (see the black rectangles in Figure 8 (b) and (c)); more importantly, the flantess-based approach using piecewise polynomials in [6] can not handle all the safety constraints considered in this paper out of the box and required modifications. The OpenOCL-based approach generates trajectories that satisfy the safety constraints but is significantly slower than other approaches.*

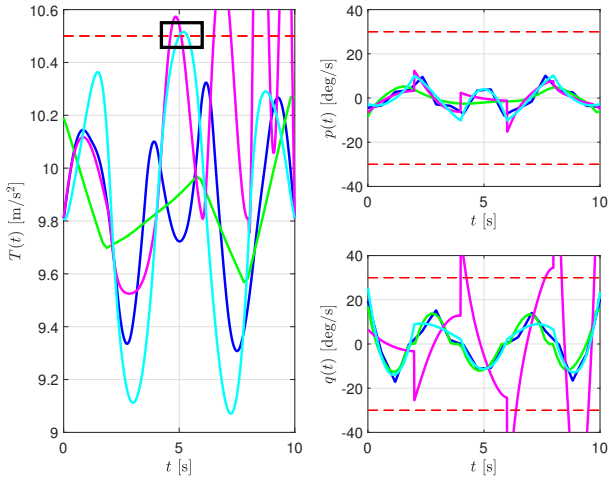
**Example 3.** *This example demonstrates the online replanning capability of the proposed SOCP-based approach. We let the quadcopter take off from a moving platform, fly through a hoop located between two waypoints at low-speed, and then land in the same moving platform. The platform is a Turtlebot driven by a human, and its position is not known a priori but measured (by the motion capture system) online and utilized in the SOCP-based trajectory generation. The reference trajectory is recomputed by solving* (FLAT-SOCP) *in real-time, where the initial and final positions of the reference trajectory are both set to be the position of the moving platform. The tracking controller is obtained by solving* (CBF-QP) *online. The safety*




(a) The position trajectories with solve time displayed in seconds.




(b) The roll $\phi$ and pitch $\theta$ trajectories. The black rectangles highlight instances when the flatness-based approach using piecewise polynomials in [6] fails to provide safety due to the constraints being enforced only at discrete times.

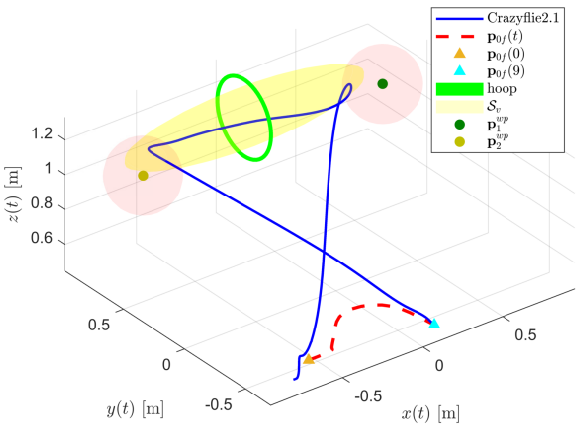



(c) The input $\mathbf{u}$ trajectories.

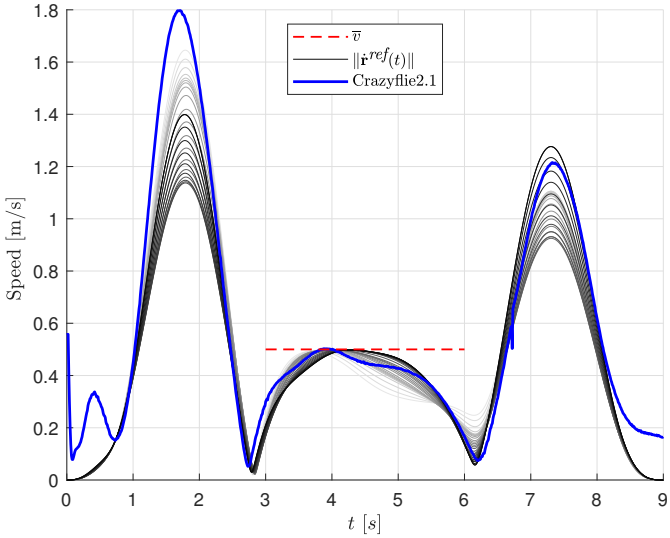
Fig. 8. Trajectories obtained from the simulation of the four approaches considered in Example 2.

*specifications for the reference trajectory are:*

$$\mathbf{r}^{ref}(t) \in \mathcal{S}_v, \quad \|\dot{\mathbf{r}}^{ref}(t)\| \leq \overline{v} = 0.5, \quad t \in [3, 6), \quad \text{(46a)}$$




(a) Scenario for Example 3 showing the waypoints $\mathbf{p}_1^{wp}$ and $\mathbf{p}_2^{wp}$ with radius $d_{wp} = 0.2$, the position constraint set $\mathcal{S}_v$ to clear the *hoop* obstacle, and the measured trajectories of the dynamic platform (i.e., $\mathbf{p}_{0f}(t)$) and Crazyflie2.1. The quadcopter takes off near $\mathbf{p}_{0f}(0)$ and lands near $\mathbf{p}_{0f}(9)$, which is accomplished by recomputing the reference trajectory $\mathbf{r}^{\text{ref}}(t)$.




(b) Velocity profiles of all trajectories recomputed. Profiles computed at the beginning of the experiment ($t \approx 0$) are traced in light grey and those computed at the end of the experiment ($t \approx 9$) are traced in darker grey. All computed reference trajectories satisfy the safety specification $\|\dot{\mathbf{r}}^{\text{ref}}(t)\|_2 \leq 0.5$ for $t \in [3, 6)$. Also shown is the speed of the Crazyflie2.1 as it tracks the evolving reference trajectory.

Fig. 9. Visualization of the experimental data for Example 3.

$$\mathbf{r}^{ref}(0) = \mathbf{r}^{ref}(9) = \mathbf{p}_{0f}(t), \quad \dot{\mathbf{r}}^{ref}(0) = \dot{\mathbf{r}}^{ref}(9) = 0, \quad \text{(46b)}$$

$$\|\mathbf{r}^{ref}(2.5) - \mathbf{p}_1^{wp}\|_2 \leq d_{wp}, \|\mathbf{r}^{ref}(6.5) - \mathbf{p}_2^{wp}\|_2 \leq d_{wp}, \quad \text{(46c)}$$

*where* $\mathbf{p}_{0f}(t)$ *is the position of the platform at time* $t$, $\mathbf{p}_1^{wp} = (0.75, 0.6, 1.1)^T$ *and* $\mathbf{p}_2^{wp} = (-0.75, 0.6, 1.1)^T$ *are the waypoints with desired radius* $d_{wp} = 0.2$, *and the set* $\mathcal{S}_v$ *is defined as* $\mathcal{S}_v \triangleq \{\mathbf{r} \in \mathbb{R}^3 : \|A_s\mathbf{r} + \mathbf{b}_s\|_2 \leq 1\}$, *with* $A_s = diag(1.33, 13.3, 13.3)$ *and* $\mathbf{b}_s = (0, -10, -14.7)^T$, *which describes an ellipsoid centered at* $\mathbf{p}_h = (0, 0.75, 1.1)^T$ *to ensure the quadcopter clears the* hoop *obstacle.*

*We begin by considering a B-spline curve* $\mathbf{r}^{ref}(t)$ *as defined in* (5) *with* $N = 45$, $d = 5$, $\tau_0 = 0$ *and* $\tau_v = 9$. *Including*

(46b) *and* (46c) *in* (FLAT-SOCP) *is straightforward. However,* (46a) *indicates position and velocity bounds over a local time interval and will result in constraints over a subset of the control points and VCPs. Since* $\boldsymbol{\tau}$ *is clamped and uniform, we have that* $[3,6) \in [\tau_{18}, \tau_{33}) = [2.85, 6.15)$*. Using Proposition 1, we obtain the following constraints on the control points and* $1st$ *order VCPs:*

$$\mathbf{P}_i \in S_v,\ i = 13,\ldots,32, \quad \|\mathbf{P}_j^{(1)}\|_2 \leq \overline{v},\ j = 14,\ldots,32.$$

*In the experiment, measurements of* $\mathbf{p}_{0f}(t)$ *are obtained at 100 Hz and the convexity of* (FLAT-SOCP) *allows us to recompute trajectories* $\mathbf{r}^{ref}(t)$ *satisfying* (46) *at 30Hz. The tracking is done by a nominal controller* $\pi(\mathbf{z})$ *filtered by* (CBF-QP) *with parameters* $\delta = 0.1$*,* $a_1 = 6$ *and* $a_2 = 8$*.*

*We implement this experiment using a Crazyflie2.1 nano quadcopter and a Turtlebot 2 as the dynamic platform. Figure 9(a) shows the considered scenario, the quadcopter tracking data and the measurements of* $\mathbf{p}_{0f}(t)$*. The Crazyflie2.1 takes off near* $\mathbf{p}_{0f}(0)$ *and lands near* $\mathbf{p}_{0f}(9)$*, by tracking the reference trajectory* $\mathbf{r}^{ref}(t)$*, which is recomputed at 30Hz. Additionally, we show the velocity profiles* $\|\dot{\mathbf{r}}^{ref}(t)\|_2$ *of the reference trajectories in Figure 9(b). It can be seen that all computed trajectories respect the bounded-velocity safety specification* $\|\dot{\mathbf{r}}^{ref}(t)\|_2 \leq 0.5$ *for all* $t \in [3,6)$*. We also show the measured velocity magnitude of the quadcopter throughout the experiment.*

**Example 4.** *The last example demonstrates safe trajectory planning and tracking in a cluttered environment. The definitions of the start/end zones* $\mathcal{C}_1,\ldots,\mathcal{C}_4$*, cuboid obstacles* $\mathcal{O}_1, \mathcal{O}_2$*,* hoop $\mathcal{O}_3$*,* desk $\mathcal{O}_4$*,* ladder $\mathcal{O}_5$*, safe flight space* $\mathcal{F}$*, and the convex sets* $\mathcal{S}_1,\ldots,\mathcal{S}_6$ *(chosen as in Proposition 2) are described in Appendix B. The environment configuration is shown in Figure 10. We note that this environment was setup and measured offline prior to the experiment; thus, the quadcopter effectively has access to a "map" of its environment during trajectory replanning. We design trajectories that avoid the obstacles and take the quadcopter from any* $\mathbf{p}^0 \in \mathcal{C}_i$ *to any* $\mathbf{p}^f \in \mathcal{C}_j$ *such that* $\mathbf{r}^{ref}(t) \in \mathcal{F}$ *for all* $t \in [t_0, t_f]$*,* $\mathbf{r}^{ref}(t_0) = \mathbf{p}^0$ *and* $\mathbf{r}^{ref}(t_f) = \mathbf{p}^f$*. Because of the choice of convex sets* $\{\mathcal{S}_j\}$*, given start zone* $\mathcal{C}_i$ *and end zone* $\mathcal{C}_j$*, we can always find a finite sequence* ($n_s < \infty$) *of indices* $(k_1, k_2, \ldots, k_{n_s})$*,* $k_l \in \{1,\ldots,6\}$*,* $l = 1,\ldots,n_s$ *such that* $(\mathcal{C}_i \cup \mathcal{S}_{k_1} \cup \mathcal{S}_{k_2} \cup \ldots \cup \mathcal{S}_{k_{n_s}} \cup \mathcal{C}_j) \subset \mathcal{F}$*,* $\mathcal{C}_i \cap \mathcal{S}_{k_1} \neq \emptyset$*,* $\mathcal{S}_{k_{n_s}} \cap \mathcal{C}_j \neq \emptyset$ *and* $\mathcal{S}_{k_l} \cap \mathcal{S}_{k_{l+1}} \neq \emptyset$ *with* $l = 1,\ldots,n_s - 1$*, where* $i, j \in \{1,\ldots,4\}$ *are the indices of the start zone* $\mathcal{C}_i$ *and the end zone* $\mathcal{C}_j$*, respectively. The sequence of sets* $(\mathcal{S}_{k_1},\ldots,\mathcal{S}_{k_{n_s}})$ *satisfies the conditions of Proposition 2 and we can use it to obtain constraints for* (FLAT-SOCP) *such that its solution results in a B-spline curve* $\mathbf{r}^{ref}(t)$ *that safely navigates the cluttered environment.*

*In the experiment, we set* $\mathbf{p}^0 \in \mathcal{C}_i$ *as the current position of the quadcopter, and choose a random point* $\mathbf{p}^f \in \mathcal{C}_j$ *as the next goal position where* $\mathcal{C}_j \in \{1,\ldots,4\}$ *is also randomly picked. Then, we find a sequence of convex sets* $(\mathcal{S}_{k_1},\ldots,\mathcal{S}_{k_{n_s}})$ *as above and obtain constraints for* (FLAT-SOCP) *from Proposition 2. Finally, we solve the optimization problem*

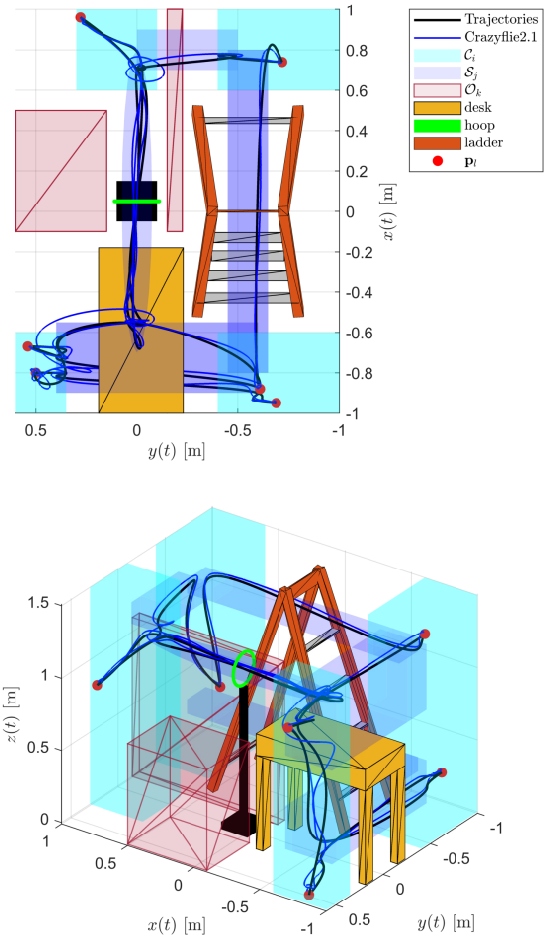



Fig. 10. Experimental setup of the cluttered environment and the trajectories generated for Example 4. The designated takeoff/landing position is $\mathbf{p}_1 \in \mathcal{C}_2$. The other points $\mathbf{p}_l, l = 2,\ldots,6$ are chosen randomly from the zones $\mathcal{C}_i, i = 1,\ldots,4$. The sequence of points visited by the quadcopter is $\mathbf{p}_1 \in \mathcal{C}_2, \mathbf{p}_2 \in \mathcal{C}_3, \mathbf{p}_3 \in \mathcal{C}_2, \mathbf{p}_4 \in \mathcal{C}_1, \mathbf{p}_5 \in \mathcal{C}_3, \mathbf{p}_6 \in \mathcal{C}_4, \mathbf{p}_1 \in \mathcal{C}_2$. The reference trajectories are shown in black and the real trajectory of the recorded tracking data of the Crazyflie 2.1 quadcopter is shown in blue. We emphasize that the trajectories were generated on-the-fly as each end point was drawn after completing the tracking of the previous trajectory. The average solve-time of (FLAT-SOCP) was 0.0135 seconds (see Remark 10). The supplementary video is given in [56].

(FLAT-SOCP) *and generate a reference trajectory. When the quadcopter finishes tracking the trajectory using the CBF-QP-based controller, the process is repeated for a new random* $\mathbf{p}^f \in \mathcal{C}_k, k \in \{1,\ldots,4\}$*. We repeat this process 6 times, computing each trajectory on-the-fly. The result is a smooth, continuous trajectory that safely navigates the cluttered environment, shown in Figure 10. In the experiment, we forced the last* $\mathbf{p}^f$ *to match the takeoff location of the quadcopter. The resulting sequence of points visited by the quadcopter was* $\mathbf{p}_1 \in \mathcal{C}_2, \mathbf{p}_2 \in \mathcal{C}_3, \mathbf{p}_3 \in \mathcal{C}_2, \mathbf{p}_4 \in \mathcal{C}_1, \mathbf{p}_5 \in \mathcal{C}_3, \mathbf{p}_6 \in \mathcal{C}_4, \mathbf{p}_1 \in \mathcal{C}_2$*, and* $\mathbf{p}_l, l = 1,\ldots,6$ *are shown in Figure 10.*

**Remark 10.** *For experiments that require online replanning, we use Yalmip's* `optimizer` *objects to reduce the overhead compiling time, which greatly improves solve time of parametric problems. Typical compile times for an* `optimizer`

*object of* (FLAT-SOCP) *are about* 0.7 *seconds and, as previously stated, any subsequent solving of the parameterized* (FLAT-SOCP) *takes between* 0.01 *and* 0.033 *seconds.*

## VIII. Conclusions

We presented a convex optimization-based framework that achieves rigorous continuous-time safety for the trajectory planning and tracking of quadcopters. We also showed several experimental examples that demonstrate the usefulness of the presented framework. We highlight that the presented trajectory optimization method can easily generalize to other differentially flat systems, especially when geometric interpretation of the flat-space constraints can be inferred. The presented safe tracking approach can also generalize to other control-affine mechanical systems. In future work, we will explore customized solvers for the SOCP problem involved, and incorporate high-level "safe corridor" finding algorithms into the proposed framework. We also plan to extend the proposed approach to real-world scenarios considering sensor noise and external disturbances and to achieve perception-based, safe navigation and tracking.

## References


[1] G. Rousseau, C. S. Maniu, S. Tebbani, M. Babel, and N. Martin, "Minimum-time b-spline trajectories with corridor constraints. application to cinematographic quadrotor flight plans," *Control Eng. Pract.*, vol. 89, pp. 190–203, 2019.

[2] L. Apvrille, T. Tanzi, and J.-L. Dugelay, "Autonomous drones for assisting rescue services within the context of natural disasters," in *31st URSI Gen. Assem. Sci. Symp. (URSI GASS)*. IEEE, 2014, pp. 1–4.

[3] J. Navia, I. Mondragon, D. Patino, and J. Colorado, "Multispectral mapping in agriculture: Terrain mosaic using an autonomous quadcopter UAV," in *Int. Conf. Unmanned Aircr. Syst. (ICUAS)*. IEEE, 2016, pp. 1351–1358.

[4] S. Tang, V. Wüest, and V. Kumar, "Aggressive flight with suspended payloads using vision-based control," *IEEE Rob. Autom. Lett.*, vol. 3, no. 2, pp. 1152–1159, 2018.

[5] A. L. Dontchev, M. Huang, I. V. Kolmanovsky, and M. M. Nicotra, "Inexact newton–kantorovich methods for constrained nonlinear model predictive control," *IEEE Trans. Autom, Control*, vol. 64, no. 9, pp. 3602–3615, 2018.

[6] D. Mellinger and V. Kumar, "Minimum snap trajectory generation and control for quadrotors," in *IEEE Int. Conf. Rob. Autom. (ICRA)*, 2011, pp. 2520–2525.

[7] C. Richter, A. Bry, and N. Roy, "Polynomial trajectory planning for aggressive quadrotor flight in dense indoor environments," in *Robotics Research*. Springer, 2016, pp. 649–666.

[8] D. Dueri, Y. Mao, Z. Mian, J. Ding, and B. Açikmeşe, "Trajectory optimization with inter-sample obstacle avoidance via successive convexification," in *IEEE 56th Annu. Conf. Decis. Control (CDC)*, 2017, pp. 1150–1156.

[9] Y. Mao, M. Szmuk, X. Xu, and B. Açikmese, "Successive convexification: A superlinearly convergent algorithm for non-convex optimal control problems," *arXiv preprint arXiv:1804.06539*, 2018.

[10] J. F. Bonnans and A. Festa, "Error estimates for the Euler discretization of an optimal control problem with first-order state constraints," *SIAM J. Numer. Anal.*, vol. 55, no. 2, pp. 445–471, 2017.

[11] A. L. Dontchev, "Error estimates for a discrete approximation to constrained control problems," *SIAM J. Numer. Anal.*, vol. 18, no. 3, pp. 500–514, 1981.

[12] F. Stoican, I. Prodan, D. Popescu, and L. Ichim, "Constrained trajectory generation for UAV systems using a B-spline parametrization," in *25th Mediterr. Conf. Control Autom. (MED)*. IEEE, 2017, pp. 613–618.

[13] N. T. Nguyen, I. Prodan, and L. Lefèvre, "Flat trajectory design and tracking with saturation guarantees: a nano-drone application," *Int. J. Control*, vol. 93, no. 6, pp. 1266–1279, 2020.

[14] F. Stoical, V.-M. Ivănuşcă, I. Prodan, and D. Popescu, "Obstacle avoidance via B-spline parametrizations of flat trajectories," in *24th Mediterr. Conf. Control Autom. (MED)*. IEEE, 2016, pp. 1002–1007.

[15] L. Tang, H. Wang, Z. Liu, and Y. Wang, "A real-time quadrotor trajectory planning framework based on B-spline and nonuniform kinodynamic search," *J. Field Rob.*, vol. 38, no. 3, pp. 452–475, 2021.

[16] D. Invernizzi, M. Giurato, P. Gattazzo, and M. Lovera, "Comparison of control methods for trajectory tracking in fully actuated unmanned aerial vehicles," *IEEE Trans. Control Syst. Technol.*, vol. 29, no. 3, pp. 1147–1160, 2020.

[17] E. Tal and S. Karaman, "Accurate tracking of aggressive quadrotor trajectories using incremental nonlinear dynamic inversion and differential flatness," *IEEE Trans. Control Syst. Technol.*, vol. 29, no. 3, pp. 1203–1218, 2020.

[18] P. Foehn and D. Scaramuzza, "Onboard state dependent LQR for agile quadrotors," in *IEEE Int. Conf. Rob. Autom. (ICRA)*, 2018, pp. 6566–6572.

[19] M. Greeff and A. P. Schoellig, "Flatness-based model predictive control for quadrotor trajectory tracking," in *IEEE/RSJ Int. Conf. Intell. Rob. Syst. (IROS)*, 2018, pp. 6740–6745.

[20] S. Lupashin, A. Schöllig, M. Sherback, and R. D'Andrea, "A simple learning strategy for high-speed quadrocopter multi-flips," in *IEEE Int. Conf. Rob. Autom. (ICRA)*, 2010, pp. 1642–1648.

[21] A. Singla, S. Padakandla, and S. Bhatnagar, "Memory-based deep reinforcement learning for obstacle avoidance in UAV with limited environment knowledge," *IEEE Trans. Intell. Transp. Syst.*, 2019.

[22] F. Berkenkamp, M. Turchetta, A. P. Schoellig, and A. Krause, "Safe model-based reinforcement learning with stability guarantees," *Adv. Neural Inf. Process. Syst. (NeurIPS)*, vol. 2, pp. 909–919, 2018.

[23] A. D. Ames, X. Xu, J. W. Grizzle, and P. Tabuada, "Control barrier function based quadratic programs for safety critical systems," *IEEE Trans. Autom. Control*, vol. 62, no. 8, pp. 3861–3876, 2017.

[24] X. Xu, P. Tabuada, J. W. Grizzle, and A. D. Ames, "Robustness of control barrier functions for safety critical control," *IFAC-PapersOnLine*, vol. 48, no. 27, pp. 54–61, 2015.

[25] L. Wang, E. A. Theodorou, and M. Egerstedt, "Safe learning of quadrotor dynamics using barrier certificates," in *IEEE Int. Conf. Rob. Autom. (ICRA)*, 2018, pp. 2460–2465.

[26] D. Zheng, H. Wang, J. Wang, X. Zhang, and W. Chen, "Toward visibility guaranteed visual servoing control of quadrotor UAVs," *IEEE/ASME Trans. Mechatron.*, vol. 24, no. 3, pp. 1087–1095, 2019.

[27] B. Xu and K. Sreenath, "Safe teleoperation of dynamic UAVs through control barrier functions," in *IEEE Int. Conf. Rob. Autom. (ICRA)*, 2018, pp. 7848–7855.

[28] V. Cichella, I. Kaminer, C. Walton, and N. Hovakimyan, "Optimal motion planning for differentially flat systems using Bernstein approximation," *IEEE Control Syst. Lett.*, vol. 2, no. 1, pp. 181–186, 2017.

[29] M. Hehn and R. D'Andrea, "Quadrocopter trajectory generation and control," *IFAC Proc. Vol.*, vol. 44, no. 1, pp. 1485–1491, 2011.

[30] J. Koenemann, G. Licitra, M. Alp, and M. Diehl, "OpenOCL–open optimal control library," 2017.

[31] M. W. Mueller, M. Hehn, and R. D'Andrea, "A computationally efficient motion primitive for quadrocopter trajectory generation," *IEEE Trans. Rob.*, vol. 31, no. 6, pp. 1294–1310, 2015.

[32] M. J. Van Nieuwstadt and R. M. Murray, "Real-time trajectory generation for differentially flat systems," *Int. J. Robust Nonlinear Control: IFAC-Affiliated Journal*, vol. 8, no. 11, pp. 995–1020, 1998.

[33] H. Nguyen, M. Kamel, K. Alexis, and R. Siegwart, "Model predictive control for micro aerial vehicles: A survey," in *Eur. Control Conf. (ECC)*. IEEE, 2021, pp. 1556–1563.

[34] D. Malyuta, Y. Yu, P. Elango, and B. Açıkmeşe, "Advances in trajectory optimization for space vehicle control," *Annu. Rev. Control*, vol. 52, pp. 282–315, 2021.

[35] L. Campos-Macías, D. Gómez-Gutiérrez, R. Aldana-López, R. de la Guardia, and J. I. Parra-Vilchis, "A hybrid method for online trajectory planning of mobile robots in cluttered environments," *IEEE Rob. Autom. Lett.*, vol. 2, no. 2, pp. 935–942, 2017.

[36] "PX4 open source project." [Online]. Available: https://docs.px4.io/main/en/

[37] M. Fliess, J. Lévine, P. Martin, and P. Rouchon, "Flatness and defect of non-linear systems: introductory theory and examples," *Int. J. Control*, vol. 61, no. 6, pp. 1327–1361, 1995.

[38] D. Zhou and M. Schwager, "Vector field following for quadrotors using differential flatness," in *IEEE Int. Conf. Rob. Autom. (ICRA)*, 2014, pp. 6567–6572.

[39] C. De Boor and C. De Boor, *A Practical Guide to Splines*. Springer-Verlag New York, 1978, vol. 27.

[40] F. Suryawan, "Constrained trajectory generation and fault tolerant control based on differential flatness and B-splines," Ph.D. dissertation, The University of Newcastle, Australia, 2012.

[41] X. Xu, J. W. Grizzle, P. Tabuada, and A. D. Ames, "Correctness guarantees for the composition of lane keeping and adaptive cruise control," *IEEE Trans. Autom. Sci. Eng.*, vol. 15, no. 3, pp. 1216–1229, 2018.

[42] X. Xu, "Constrained control of input–output linearizable systems using control sharing barrier functions," *Automatica*, vol. 87, pp. 195–201, 2018.

[43] M. S. Lobo, L. Vandenberghe, S. Boyd, and H. Lebret, "Applications of second-order cone programming," *Linear Algebra and its Appl*, vol. 284, no. 1-3, pp. 193–228, 1998.

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

[45] MOSEK, *The MOSEK optimization toolbox for MATLAB manual. Version 9.3*, 2021.

[46] N. T. Nguyen, I. Prodan, and L. Lefèvre, "Effective angular constrained trajectory generation for thrust-propelled vehicles," in *Eur. Control Conf. (ECC)*. IEEE, 2018, pp. 1833–1838.

[47] B. Açıkmeşe and L. Blackmore, "Lossless convexification of a class of optimal control problems with non-convex control constraints," *Automatica*, vol. 47, no. 2, pp. 341–347, 2011.

[48] M. W. Mueller and R. D'Andrea, "A model predictive controller for quadrocopter state interception," in *Eur. Control Conf. (ECC)*. IEEE, 2013, pp. 1383–1389.

[49] F. Gao, W. Wu, Y. Lin, and S. Shen, "Online safe trajectory generation for quadrotors using fast marching method and bernstein basis polynomial," in *IEEE Int. Conf. Rob. Autom. (ICRA)*, 2018, pp. 344–351.

[50] A. Hornung, K. M. Wurm, M. Bennewitz, C. Stachniss, and W. Burgard, "OctoMap: An efficient probabilistic 3D mapping framework based on octrees," *Auton. Rob.*, vol. 34, pp. 189–206, 2013.

[51] G. Tang, W. Sun, and K. Hauser, "Time-optimal trajectory generation for dynamic vehicles: A bilevel optimization approach," in *IEEE/RSJ Int Conf. Intell. Rob. Syst. (IROS)*, 2019, pp. 7644–7650.

[52] W. Sun, G. Tang, and K. Hauser, "Fast UAV trajectory optimization using bilevel optimization with analytical gradients," in *Am. Control Conf. (ACC)*. IEEE, 2020, pp. 82–87.

[53] B. Stellato, G. Banjac, P. Goulart, A. Bemporad, and S. Boyd, "OSQP: an operator splitting solver for quadratic programs," *Math. Program. Comput.*, vol. 12, no. 4, pp. 637–672, 2020.

[54] Y. Zhang, S. Walters, and X. Xu, "Control barrier function meets interval analysis: Safety-critical control with measurement and actuation uncertainties," *Am. Control Conf. (ACC)*, pp. 3814–3819, 2022.

[55] J. Löfberg, "Yalmip : A toolbox for modeling and optimization in matlab," in *Proc. Comput. aided Control Syst. Des. (CACSD)*, Taipei, Taiwan, 2004.

[56] UW ARC Lab, "Supplementary video," https://xu.me.wisc.edu/wp-content/uploads/sites/1196/2021/10/continuous-safety.mp4, 2021.


## APPENDIX

### A. Construction of Matrix $B_r$

The matrix $B_r \in \mathbb{R}^{(N+1)\times(N+r+1)}$ is only defined when $0 \le r \le d$, and can be computed from two matrices $M_{d,d-r}$ and $C_r$ such that $B_r = M_{d,d-r}C_r$; see [40] for more details. The matrix $M_{d,d-r} \in \mathbb{R}^{(N+1)\times(N-r+1)}$ can be constructed recursively by:

$$M_{d,d-r} = \begin{cases} I_{N+1}, & r = 0, \\ \prod_{i=1}^{r} f_M(\boldsymbol{\tau}, d, N, i), & 1 \le r \le d, \end{cases}$$

where each iteration in the product is a right-multiplication and the matrix-valued function $f_M : \mathbb{R}^{v+1} \times \mathbb{Z}_{>0} \times \mathbb{N} \times \mathbb{N} \to \mathbb{R}^{(N-i+2)\times(N-i+1)}$ is defined as

$$f_M(\boldsymbol{\tau}, d, N, i) = \begin{bmatrix} -a_0 & \dots & 0 \\ a_0 & \ddots & \vdots \\ \vdots & \ddots & -a_{N-i} \\ 0 & \dots & a_{N-i} \end{bmatrix},$$

where $a_k = (d-i+1)/(\tau_{k+d+1} - \tau_{k+i})$. The matrix $C_r$ is constructed as $C_r = \begin{bmatrix} \mathbf{0}_{(N+1-r)\times r} & I_{N+1-r} & \mathbf{0}_{(N+1-r)\times r} \end{bmatrix} \in \mathbb{R}^{(N-r+1)\times(N+r+1)}$ for $r \in \mathbb{Z}_{>0}$ and $C_0 = I_{N+1}$.

### B. Configuration of the Environment for Example 4

Zones $\mathcal{C}_1, \dots, \mathcal{C}_4$ are defined by $\mathcal{C}_i \triangleq \{\mathbf{r} \mid A\mathbf{r} \le \mathbf{b}_i^c\}$ where $A = \begin{bmatrix} I_3 & -I_3 \end{bmatrix}^T$, $\mathbf{b}_1^c = (1, 0.3, 1.5, -0.6, 0.1, 0)^T$, $\mathbf{b}_2^c = (-0.6, 0.6, 1.5, 1, -0.35, 0)^T$, $\mathbf{b}_3^c = (-0.6, -0.4, 1.5, 1, 1, 0)^T$ and $\mathbf{b}_4^c = (1, -0.4, 1.5, -0.6, 1, 0)^T$. Obstacles $\mathcal{O}_1, \mathcal{O}_2$ are defined by $\mathcal{O}_i \triangleq \{\mathbf{r} \mid A\mathbf{r} \le \mathbf{b}_i^o\}, i = 1, 2$, where $A$ is the same as above, $\mathbf{b}_1^o = (0.5, 0.6, 0.7, 0.1, -0.15, 0)^T$, and $\mathbf{b}_2^o = (0.1, -0.225, 1.1, 0.1, 0.15, 0)^T$. The obstacle $\mathcal{O}_3$ is a hoop centered at $(0, 0, 1.1)$ through which the quadcopter can pass; the obstacle $\mathcal{O}_4$ is a desk between $\mathcal{C}_2$ and $\mathcal{C}_3$ so that the quadcopter must fly above or below the *desk* to avoid it; the obstacle $\mathcal{O}_5$ a ladder between $\mathcal{C}_3$ and $\mathcal{C}_4$ so that the quadcopter must fly between two of its rungs. The safe flight space is defined as $\mathcal{F} = \{\mathbf{r} \mid (-1, -1, 0)^T \le \mathbf{r} \le (1, 0.6, 1.5)^T\} \setminus (\mathcal{O}_1 \cup \dots \cup \mathcal{O}_5)$. The six convex sets of the safe corridor are chosen as $\mathcal{S}_j \triangleq \{\mathbf{r} \mid A\mathbf{r} \le \mathbf{b}_j^s\} (j = 1, \dots, 5)$ where $A$ is the same as above, $\mathbf{b}_1^s = (-0.55, 0.4, 0.5, 0.9, 0.6, -0.2)^T$, $\mathbf{b}_2^s = (-0.55, 0.4, 1.3, 0.9, 0.6, -0.85)^T$, $\mathbf{b}_3^s = (0.8, -0.45, 1.3, 0.8, 0.65, -1.2)^T$, $\mathbf{b}_4^s = (0.8, -0.45, 0.47, 0.8, 0.65, -0.37)^T$, $\mathbf{b}_5^s = (0.9, 0, 1.3, -0.7, 0.5, -1.15)^T$, and $\mathcal{S}_6 \triangleq \{\mathbf{r} \mid \|A_6^s\mathbf{r} + \mathbf{b}_6^s\| \le 1\}$ with $A_6^s = \mathrm{diag}(1.33, 13.3, 13.3)$ and $\mathbf{b}_6^s = (-0.067, 0, -14.67)^T$.