2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
Madrid, Spain, October 1-5, 2018

Optimal constrained trajectory generation for quadrotors through smoothing splines

Shupeng Lai[1], Menglu Lan[2] and Ben M. Chen[1]

***Abstract*— In this paper, we present a trajectory generation method for quadrotors based on the optimal smoothing B-spline. Compared to existing methods which rely on polynomial splines or time optimal control techniques, our method systematically addresses the issue of axes-coupled and interval-wise constraints. These constraints can be used to construct safe flying zones and satisfy vehicle's physical limits. The proposed approach has also been extended to generate trajectories from the nominal plan which consists of not only points but also lines and planes, opening a door for new improvements and applications. Moreover, a closed-form solution can be obtained for cases without inequality constraints. Such a solution is numerically stable for the large-scale fitting problem, which allows us to directly fit the human sketching input from the touch device and capture all subtle details. Our approach is verified by various real flight experiments.**

I. Introduction

The trajectory generation for a quadrotor has become increasingly important with the new challenges such as human vehicle interaction [1], operating within safety zone [2] and satisfying the vehicle's feasibility conditions[3]. The quadrotors are considered as differential-flat holonomic vehicles [4]. The state trajectory can be described by its flat outputs, namely the position and yaw angle of the vehicle. Therefore, the trajectory generation is often considered as an optimization problem with an integrator chain based vehicle model. However, due to the high order nature of its system dynamics, it is not an easy task to project various design targets into the optimization problem and remain an efficiently solvable formulation. A list of frequently encountered requirements is summarized as follows.

- **Waypoint navigating:** The trajectory is required to navigate through a series of waypoints [4] . These waypoints might comes from a higher level path planner [5] or directly from the user inputs [1] as design specifications.
- **Trajectory smoothness:** Unnecessary aggressive maneuvers shall be penalized. Minimum snap trajectory [4] is widely adopted as it directly relates to the motors' input. Additionally, minimum jerk trajectory is also utilized as it appears more pleasant to a human [1].
- **Safe zone constraint:** The trajectory shall travel within certain safe regions to avoid collision [6], [2]. The result can be extended to multi-agents by separating the agents through hyper planes [7], [8].
- **Feasibility condition:** The trajectory shall satisfy limits including the maximum velocity, thrust and body rate which can be fulfilled by constraining the trajectory's derivatives [3].

In most of the current approaches, these design targets are either effective only at point-wise [4], [5], [2] or decoupled into each individual axes [6], [3], [9]. Point-wise effectiveness does not guarantee the satisfaction of constraints in between samples [7]. Whereas the axes-decoupling decreases the quality of the solution; for example, it forces the safe zone to be constructed by axis aligned cubes [6] which is shown less efficient when obstacles are not rectangular-shaped [2]. Recently in [7], [8], the Bézier curve is used to address the axes-coupled constraint during a continuous time-interval. However, it is limited to the safe zone constraints only.

In this paper, we introduce a trajectory generation method for quadrotors based on the smoothing B-spline [10], [11] which is a generalized version of the Bézier curve. It unifies several previous methods and addresses all listed design targets in a single quadratic programming (QP) problem. The advantages include:

- Systematically addressing the issue of interval-wise and axes-coupled constraints on the trajectory and its derivatives. With our method, a larger constraint volume can be constructed which increases the solution quality.
- Minimizing the trajectory's deviation to desired lines and planes over a continuous time-interval. One particular application is to suppress the trajectory's excursion between waypoints due to poor time allocations [12].
- An adjustable problem size. The number of variables in the resulting QP can be adjusted independently without increasing the order of the spline, which makes it possible to trade off between the solution quality and the computational cost.
- Numerical stability and efficiency. An efficient closed-form solution is available if no inequality constraints is needed. The proposed method is numerically stable to handle problems with thousands of densely located waypoints which enables the direct fitting of user sketching inputs from touch devices.

The rest of the paper is organized as follows. In Sec.II, related works on trajectory generation for quadrotors are reviewed. The discussion on the trajectory's mathematical representation has been covered in Sec.III; whereas in Sec.IV, we present the proposed method in detail. Results and analysis of real flight experiments are provided in Sec.V. Finally, a conclusion is drawn in Sec.VI.

[1]Shupeng Lai (elelais@nus.edu.sg) and Ben M. Chen are with the Department of Electrical and Computer Engineering, National University of Singapore (NUS).

[2]Menglu Lan is with the Graduate School for Integrative Science & Engineering, NUS.

II. Related works

The literature on trajectory generation for quadrotors are extensive. In [13], the authors proposed an efficient algorithm to handle the state-to-state transition problem. The trajectory is expressed as a polynomial and the corresponding closed form solution to the boundary value problem is found. The solution is then examined by a verification process to determine the feasibility w.r.t input constraints, and an axis decoupled convex constraint. In [14], [3], a realtime bang-zero-bang based time-optimal control (TOC) strategy is adopted to explicitly consider input and state constraints. The algorithm assumes a triple integrator model for each of the vehicle's positional axis and the trajectory problem is reformulated through a decoupling approach. A similar approach in [15] shows the decoupled trajectory can be re-synchronized to a certain extend. Based on this method, motion planning algorithms were developed in [16], [17] for quadrotors to fly in cluttered environments.

On the other hand, the TOC based approach cannot handle other optimization targets such as minimum energy and usually generates unnecessary aggressive maneuvers. A different strategy that generates minimum snap trajectory while interpolating a series of waypoints has been proposed in [4]. It used piecewise polynomials to describe the trajectory and formulated a QP problem with continuity equality constraints. This method is later improved in [5] by choosing a different set of optimization variables to eliminate the continuity equality constraints. Instead of reaching waypoints exactly, the method in [1] approximated a set of user-provided waypoints as a trajectory design technique.

In real-life applications, the vehicles are usually limited to certain safe operation zones which are subsets of the state space. The authors in [6] proposed an algorithm to generate trajectories through a series of axis aligned bounding boxes. These axes-decoupled constraints were satisfied interval-wise through repeatedly adding constraint points at the violating extrema for solving the QP in the next iteration. In [2], the method was extended to handle polyhedron corridors by adding axes-coupled corridor constraints. However, these constraints were satisfied only point-wise. Moreover, the limits on vehicle's velocity, acceleration and jerk were handled in-explicitly by adjusting the time allocation. A method that satisfies interval-wise coupled corridor constraints has been shown in [7], [8]. It assigned a fixed degree Bézier curve for each convex polyhedron and combines them through continuity constraints. However, their work considered safe corridor constraints only. A B-spline formulation was proposed in [11], [10] with constraints on the trajectory and its derivatives. The formulation explored the non-negativity and partition-of-unity properties of the base function to satisfy the piece wise linear boundary over a time interval. In this paper, we further extended the idea to handle design targets involving lines, plans and polytopes.

III. Preliminaries

A. Notation

Let $\mathcal{X} \in \mathbb{R}^{d\times 1}$ be the d dimensional *work space*, where $d \in \mathbb{N}$. In this paper, we assume $d = 3$ and $\mathcal{X} = \{\mathbf{x}, \mathbf{y}, \mathbf{z}\}$ which denotes the three-dimensional euclidean space, the heading of the vehicle is planned separately.

We use the bracketed subscript to denote specific elements in a vector or a matrix, i.e. if $a \in \mathbb{R}^{m\times 1}$, then $a_{(i)}$ represents its ith element, if $A \in \mathbb{R}^{m\times n}$, then $A_{(i,j)}$ denotes its element in row i and column j, and $A_{(i,*)}$ represents the ith row, and $A_{(*,j)}$ represents the jth column. Futher, we call $\widehat{A} = [A_{(1,*)}, A_{(2,*)}, \ldots, A_{(m,*)}]^\intercal$ as the vectorized version of A and $\|a\| = \sqrt{a^\intercal a}$. The operator $\otimes$ stands for the Kronecker product, and I_d represents an identity matrix of dimension $\mathbb{R}^{d\times d}$.

B. Clamped uniform B-spline construction

In this paper, a kth order clamped uniform B-spline S_k is chosen to represent the trajectory where S_k is defined as:

$$S_k(s) = \sum_{i=0}^{M-1} c_i N_i^k(s),\ c_i, S \in \mathcal{X},\ N_i^k \in \mathbb{R}, \tag{1}$$

with N_i^k denots the basis function and c_i is the control point. We then call $C = [c_0, c_1, \ldots c_{M-1}]$, $C \in \mathbb{R}^{d\times M}$ as the control point matrix. The basis functions are defined over an knot vector $\mathcal{K} = [s_0, s_1, \ldots, s_{M+k}]$ and a path parameter s as:

$$\begin{aligned} N_i^0(s) &= \begin{cases} 1, & \text{if } s_i \le s < s_{i+1}, \\ 0, & \text{otherwise.} \end{cases} \\ N_i^j(s) &= N_A(s,i,j)N_i^{j-1}(s) + N_B(s,i,j)N_{i+1}^{j-1}(s) \end{aligned} \tag{2}$$

where

$$\begin{aligned} N_A(s,i,j) &= \begin{cases} 0, & \text{if } s_i = s_{i+j}, \\ \frac{s-s_i}{s_{i+j}-s_i}, & \text{otherwise.} \end{cases} \\ N_B(s,i,j) &= \begin{cases} 0, & \text{if } s_{i+1} = s_{i+j+1}, \\ \frac{s_{i+j+1}-s}{s_{i+j+1}-s_{i+1}}, & \text{otherwise.} \end{cases} \end{aligned} \tag{3}$$

For the spline to be clamped and uniform, there is:

$$\begin{aligned} \mathcal{K} &= [s_0, s_1 ..., s_{M+k}] \\ &= [\underbrace{0, \ldots, 0}_{k+1 \text{ times}}, 1, 2, \ldots, \underbrace{M-k, \ldots, M-k}_{k+1 \text{ times}}] \end{aligned} \tag{4}$$

In this paper, the following two properties of the basis functions are frequently used.

- Non-negativity: $N_i^k(s) \ge 0$ for all $i \in \{0, 1, \ldots, M-1\}$ and $s \in [s_0, s_{M+k}]$.
- Partition-of-unity: $\sum_{i=0}^{M-1} N_i^k(s) = 1$ for all $s \in [s_0, s_{M+k}]$.

Following [10], we can assign a linear relation between the path paramenter s and the time t as

$$\frac{s}{t} = \alpha. \tag{5}$$

And it gives $S_k(s) = S_k(\alpha t)$ where $t \in [0, T_{\text{end}}]$ and $T_{\text{end}} = \frac{M-k}{\alpha}$.

C. Method overview

In this paper, we treat the trajectory generation as an optimization problem:

$$\begin{aligned} \min \quad & E_{\text{smooth}} + \omega_{(1)}\epsilon_{\text{point}} + \omega_{(2)}\epsilon_{\text{line}} + \omega_{(3)}\epsilon_{\text{plane}} \\ \text{s.t.} \quad & \text{Boundary conditions} \\ & \text{Safe zone constraints} \\ & \text{Feasibility conditions} \end{aligned} \tag{6}$$

where $\omega \in \mathbb{R}^3$ and all of its elements are positive. E_{smooth} control the aggressiveness of the trajectory by penalizing the integration of the square of the derivatives. ϵ_{point}, ϵ_{line} and ϵ_{plane} penalize the deviation from the desired point, straight lines and planes at the corresponding time-point or time-interval respectively. The *boundary conditions* is a set of equality constraints that determines the initial and final state of the vehicle. Whereas the *safe zone constraints* and *feasibility conditions* are sets of inequality constraints that deal with the safety and the feasibility by limiting the trajectory and its derivatives in multiple convex polytopes.

IV. Smoothing spline trajectory

In this section, we show that the optimization problem in equation 6 can be expressed as a standard QP and there exists a closed form solution under certain conditions.

A. Smoothness

The smoothness of the trajectory is achieved by penalizing the derivatives of the trajectory:

$$\begin{aligned} E_{\text{smooth}} &= \sum_{n=1}^{k} \sigma_{(n)} \int_{-\infty}^{\infty} \left\| \frac{\mathrm{d}^n S_k}{\mathrm{d}t^n} \right\|^2 dt \\ &= \sum_{n=1}^{k} \sum_{i=1}^{d} \sigma_{(n)} \int_{-\infty}^{\infty} (\frac{\mathrm{d}^n S_{k(i)}}{\mathrm{d}t^n})^2 dt \end{aligned} \tag{7}$$

where $\sigma_{(n)} \geq 0$, $\forall n \in \{1, \ldots, k\}$. Following [11], it can be expressed as a quadratic form as

$$\widehat{C}^\intercal \left[\sum_{n=1}^{k} (\sigma_{(n)} V_n) \otimes I_d \right] \widehat{C} \tag{8}$$

where

$$V_{n(i,j)} = \alpha^{2n-1} \int_{-\infty}^{\infty} \frac{d^n N_i^k(s)}{ds^n} \frac{d^n N_j^k(s)}{ds^n} ds \tag{9}$$

and $\widehat{C}$ is the vectorized form of C. Since Eq. (7) is guaranteed to be equal or larger than zero, $\sigma_{(n)} V_n \otimes I_d$ is positive semi-definite. Furthermore, in [4], the snap of the trajectory is shown directly related to the control input of the quadrotor. Therefore, the integration of the square of the snap is used to evaluate the aggressiveness of the trajectory and is called the *snap cost*. The higher the *snap cost*, the more aggressive the trajectory is.

B. Waypoints approximation

Assume there are N_{W} waypoints to be approximated. The ith waypoint is defined by a position $d_i \in \mathcal{X}$ and a time-point $t_i \in \mathbb{R}$ with $i \in \{0, 1, \ldots, N_{\text{W}} - 1\}$. It indicates the location d_i is to be approached at time t_i. Our method then penalize the distance from the trajectory to the waypoints at the corresponding time-points. Let $D = [d_0, d_1, ..., d_{N_{\text{W}}-1}]^{\text{T}}$ and $T = [t_0, t_1, ..., t_{N_{\text{W}}-1}]^{\text{T}}, t_i \in \mathbb{R}$, the cost function can be written as

$$\begin{aligned} \epsilon_{\text{point}} &= \sum_{i=0}^{N_{\text{W}}-1} \|S_k(\alpha t_i) - d_i\|^2 \\ &= \sum_{i=0}^{N_{\text{W}}-1} \sum_{j=1}^{d} (S_{k(j)}(\alpha t_i) - d_{i(j)})^2 \\ &= \sum_{j=1}^{d} (HC_{(j,*)} - D_{(j,*)})^\intercal (HC_{(j,*)} - D_{(j,*)}) \\ &= \widehat{C}^\intercal (H^\intercal H \otimes I_d) \widehat{C} - 2\widehat{D}^\intercal (H \otimes I_d) \widehat{C} + \widehat{D}^\intercal \widehat{D} \end{aligned} \tag{10}$$

where $H \in \mathbb{R}^{N \times M}$ is constructed as

$$\begin{aligned} H_{(i+1,*)} = [N_0^k(\alpha t_i), N_1^k(\alpha t_i), ..., N_{M-1}^k(\alpha t_i)], \\ i \in \{0, 1, \ldots, N_{\text{W}} - 1\}. \end{aligned} \tag{11}$$

Finally, we note that $H^\intercal H \otimes I_d$ is positive-semi definite. On the other hand, if these waypoints are to be reached exactly, we can set equality constraints $HC_{(j,*)} - D_{(j,*)} = 0$, $\forall j = \{1, \ldots, d\}$.

C. Straight line approximation

We generalize the results in Section IV-B to minimize the trajectory's deviation to desired straight lines. Assume there are N_{L} desired straight lines. And the jth straight line is defined by two points $v_j, w_j \in \mathcal{X}$ and a time-interval $[\tau_j, \rho_j)$, where $j \in \{0, 1, \ldots, N_{\text{L}} - 1\}$. It indicates the trajectory shall approach the straight line passing through v_j, w_j during the time-interval $[\tau_j, \rho_j)$. Our method then penalizes the deviation from the straight line.

Given a point $p \in \mathcal{X}$, the square of its distance to the jth straight line can be written as

$$e_{\text{l}}^2 = p^\intercal Q_{\text{l}j} p + R_{\text{l}j} p + S_{\text{l}j} \tag{12}$$

where

$$\begin{aligned} Q_{\text{l}j} &= \frac{(u_j^\intercal u_j) \otimes I_d - u_j u_j^\intercal}{u_j^\intercal u_j} \\ R_{\text{l}j} &= \frac{-2u_j^\intercal u_j w_j - w_j^\intercal u_j^\intercal u_j}{u_j^\intercal u_j} \\ S_{\text{l}j} &= \frac{u_j^\intercal u_j (w_j^\intercal w_j) - w_j^\intercal (u_j u_j^\intercal) w_j}{u_j^\intercal u_j} \\ u_j &= v_j - w_j \end{aligned} \tag{13}$$

Assume c_d is the furthest control point to the straight line, we show that the trajectory will deviate from the straight line no further than c_d.

Proof: From Eq. (1) and (5), a point p on the trajectory at time t_s is:

$$p = \sum_{i=0}^{M-1} c_i N_i^k(\alpha t_s) \tag{14}$$

For simplicity, we use N_i to denote $N_i^k(\alpha t_s)$ and drop the subscripts for $Q_{\text{l}j}$,$R_{\text{l}j}$ and $S_{\text{l}j}$. Then there is

$$\begin{aligned} e_{\text{l}}^2 &= (\sum c_i N_i)^\intercal Q (\sum c_i N_i) + R(\sum c_i{}^\intercal N_i) + S \\ &= \sum_i \sum_m N_i N_m c_i{}^\intercal Q c_m + \sum_i N_i R c_i + S \end{aligned} \tag{15}$$

From Eq. (13), we notice Q is positive-semi definite, then

there is

$$c_i^\intercal Q c_m \leq \frac{1}{2}(c_i^\intercal Q c_i + c_m^\intercal Q c_m) \tag{16}$$

By the non-negativity and partition-of-uinity, there are:

$$\sum_i N_i c_i = \frac{1}{2}\sum_i \sum_m N_i N_m (c_i + c_m) \tag{17}$$

and

$$\sum_i \sum_m N_i N_m = 1 \tag{18}$$

And c_d being the furthest control point

$$c_d^\intercal Q c_d + R c_d \geq c_i^\intercal Q c_i + R c_i,\ \forall i \in \{0, 1, \ldots, M-1\}. \tag{19}$$

Substitute Eq. (16), (17), (18) and (19) into (15) gets:

$$\begin{aligned} e_{\mathrm{l}}^2 &\leq \frac{1}{2}\sum_i \sum_m N_i N_m (c_i^\intercal Q c_i + c_m^\intercal Q c_m \\ &\quad + R c_i + R c_m) + S \\ &\leq \sum_i \sum_m N_i N_m (c_d^\intercal Q c_d + R c_d) + S \\ &= c_d^\intercal Q c_d + R c_d + S \end{aligned} \tag{20}$$

■

Eq. (20) states that we could penalize the deviation of the trajectory from the desired straight line by minimizing the distance of the control points to it. Now we consider to penalize the deviation only within a certain time interval $[\tau_j, \rho_j)$. For a time-point such as τ_i, the corresponding path variable is $\alpha\tau_j$ which falls in the span of knot vector $[s_{\eta(\tau_j)}, s_{\eta(\tau_j)+1})$ with $\eta(\tau_j) = \lfloor \alpha\tau_j \rfloor + k$. From Eq. (3), it is found on a knot span $[u_i, u_{i+1})$, at most $k+1$ basis functions are non-zero, specifically: $N_{i-k}^k, N_{i-k+1}^k, \ldots, N_i^k$. Therefore, on the time interval $[\tau_j, \rho_j)$, only the basis functions $N_i^k(s)$, $i \in \{\eta(\tau_j) - k, \ldots, \eta(\rho_j)\}$ is possibly non-zero. And it is safe to ignore the control points whose corresponding basis functions are guaranteed to be zero. As they will have no effects on the spline during time interval $[\tau_j, \rho_j)$. Let $\lambda_{\tau_j} = \eta(\tau_j) - k$, $\lambda_{\rho_j} = \eta(\rho_j)$, and a mapping matrix Λ_i between the vectorized control point matrix and the ith control point:

$$c_i = \Lambda_i \widehat{C},\ i \in \{0, 1, \ldots, M-1\}. \tag{21}$$

Then the cost function to penalize the deviation to the desired N_{L} straight lines can be written as

$$\begin{aligned} \epsilon_{\mathrm{line}} &= \sum_{j=0}^{N_{\mathrm{L}}-1} \sum_{i=\lambda_{\tau_j}}^{\lambda_{\rho_j}} (c_i^\intercal Q_{\mathrm{l}j} c_i + R_{\mathrm{l}j} c_i) \\ &= \widehat{C}^\intercal \Big(\sum_{j=0}^{N_{\mathrm{L}}-1} \sum_{i=\lambda_{\tau_j}}^{\lambda_{\rho_j}} \Lambda_i^\intercal Q_{\mathrm{l}j} \Lambda_i \Big) \widehat{C} + R_{\mathrm{l}j} \Big(\sum_{j=0}^{N_{\mathrm{L}}-1} \sum_{i=\lambda_{\tau_j}}^{\lambda_{\rho_j}} \Lambda_i \Big) \widehat{C} \end{aligned} \tag{22}$$

Since $Q_{\mathrm{l}j}$ is positive-semi definite, so does the $\Lambda_i^\intercal Q_{\mathrm{l}j} \Lambda_i$.

D. Plane Approaching

We also covers the minimization of the trajectory's deviation to desired planes. Assume there are N_{P} desired planes. And the jth plane is defined by an anchor point $g_j \in \mathcal{X}$, a normal vector ζ_j and a time-interval $[\tau_j, \rho_j)$, where $j \in \{0, 1, \ldots, N_{\mathrm{P}} - 1\}$. It indicates the trajectory shall approach the plane contains g_j and perpendicular to ζ_j during the time-interval $[\tau_j, \rho_j)$. Similar to straight lines, the square of the distance from a point p to a plane can be written as

$$e_{\mathrm{p}}^2 = p^\intercal Q_{\mathrm{p}j} p + R_{\mathrm{p}j} p + S_{\mathrm{p}j} \tag{23}$$

where

$$Q_{\mathrm{p}j} = \frac{\zeta_j \zeta_j^\intercal}{\zeta_j^\intercal \zeta_j},\ R_{\mathrm{p}j} = \frac{-2 g_j^\intercal (\zeta_j \zeta_j^\intercal)}{\zeta_j^\intercal \zeta_j},\ S_{\mathrm{p}j} = \frac{g_j^\intercal (\zeta_j \zeta_j^\intercal) g_j}{\zeta_j^\intercal \zeta_j}. \tag{24}$$

Noticing $Q_{\mathrm{p}j}$ is positive-semi definite, with the results in Sec.IV-C, the cost function to penalize the deviation from the N_{P} desired planes can be written as

$$\epsilon_{\mathrm{plane}} = \widehat{C}^\intercal \Big(\sum_{j=0}^{N_{\mathrm{P}}-1} \sum_{i=\lambda_{\tau_j}}^{\lambda_{\rho_j}} \Lambda_i^\intercal Q_{\mathrm{p}j} \Lambda_i \Big) \widehat{C} + R_{\mathrm{p}j} \Big(\sum_{j=0}^{N_{\mathrm{P}}-1} \sum_{i=\lambda_{\tau_j}}^{\lambda_{\rho_j}} \Lambda_i \Big) \widehat{C} \tag{25}$$

E. Safe zone constraint

The safe zone is modeled as multiple closed convex polytopes where the interior of each one can be expressed by a linear inequal constraint. Assume there are N_{C} convex polytopes and the interior of the jth one is represented by $\{p \in \mathcal{X} | A_j p \leq b_j\}$ with $j \in \{0, 1, \ldots, N_{\mathrm{C}} - 1\}$. We now show that if all control points for a B-spline satisfy

$$A c_i \leq b,\ i \in \{0, 1, \ldots, M-1\}, \tag{26}$$

so does the entire trajectory.

Proof: Assume a point p is select from the trajectory at time t_s. For simplicity, we use N_i to denote $N_i^k(t_s)$ then there is

$$Ap = A\sum_i N_i c_i \quad = \sum_i N_i (A c_i) \tag{27}$$

By the non-negativity and partition-of-unity of the basis functions, and Eq. (26),

$$\sum_i N_i (A c_i) \leq \sum_i N_i b = b \quad \Rightarrow \quad Ap \leq b \tag{28}$$

■

If the jth polytope constraint is only effective on the time interval $[\tau_j, \rho_j)$, following the analysis in Section IV-C, it can be achieved by having $A_j c_i \leq b_j,\ \forall i \in \{\eta(\tau_j) - k, \ldots, \eta(\rho_j)\}$. Combining Eq. (21), we have the following inequality constraints for the safe zones

$$\begin{aligned} &A_j \Lambda_i \widehat{C} < b_j, \forall j \in \{0, \ldots, N_{\mathrm{C}} - 1\}, \\ &\forall i \in \{\eta(\tau_j) - k, \ldots, \eta(\rho_j)\}. \end{aligned} \tag{29}$$

F. Feasibility conditions

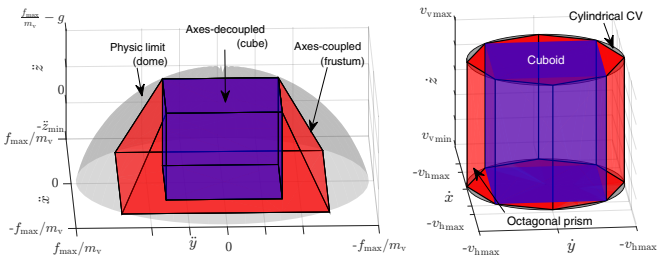
For a trajectory to be feasible, it has to satisfy the physical limits on the speed, the thrust and the body rate; as well as external restrictions such as safety and sensor capability. From [3], the thrust limits can be satisfied by:

$$\begin{aligned} \|f\| = \sqrt{\ddot{x}^2 + \ddot{y}^2 + (\ddot{z} + g)^2} &\leq \frac{f_{\max}}{m_{\mathrm{v}}} \\ \ddot{z} \geq \ddot{z}_{\min} &\geq \frac{f_{\min}}{m_{\mathrm{v}}} - g \end{aligned} \tag{30}$$

where x, y, z denotes the position in the work space, f the total thrust of the vehicle, $f_{\min}$, $f_{\max}$ the minimum and maximum available thrust, m_{v} the mass of the vehicle, g the gravity and $\ddot{z}_{\min}$ a pre-fixed design variable. And the maximum body rate $\omega_{\max}$ can be satisfied by:

$$\sqrt{\ddot{x}^2 + \ddot{y}^2 + \ddot{z}^2} \leq (\ddot{z}_{\min} + g)\omega_{\max} \tag{31}$$

Eq. (30) and (31) gives a constraint volume (CV) in the shape of a dome (Fig. 1(a)) on the acceleration and a



(a) Dome (b) Cylinder

Fig. 1. Constrained volume for acceleration.Axes-decoupled methods select an axis-aligned cuboid (the blue cube). Our axes-coupled method selects arbitrarily shaped convex polytopes (the red frustrum and octagonal prism).

sphere on the jerk respectively. To transform the CV into linear constraints, one needs to find convex polytopes in its interior. Previous methods [6], [3] only assign inequality constraints on each independent axes which results an axes-aligned cuboid (the blue cuboids in Fig. 1). Our method allows axes-coupled constraints and results in an arbitrarily shaped convex polytope which can cover a larger volume as the cuboid is just a special case of convex polytope. Other than the dome and the sphere, cylindrical CVs are also commonly considered for external restrictions like safety or sensor limits. It is intuitive and adjustable to human operators for different restrictions. The dynamical differences on the quadrotor's horizontal and vertical axes can be respected by assigning them with different limits. For example, a cylindrical CV on the velocity (Fig. 1(b)) can be expressed by

$$\begin{aligned} \sqrt{\dot{x}^2+\dot{y}^2} &\le v_{\mathrm{hmax}} \\ v_{\mathrm{vmin}} \le \dot{z} &\le v_{\mathrm{vmax}} \end{aligned} \tag{32}$$

where $v_{\mathrm{hmax}}, v_{\mathrm{vmin}}, v_{\mathrm{vmax}}$ are maximum horizontal, minimum vertical and maximum vertical accelerations respectively. Then we could define the vector $v_{\mathrm{CV}} = [v_{\mathrm{hmax}}, v_{\mathrm{vmin}}, v_{\mathrm{vmax}}]$. And similarly there are $a_{\mathrm{CV}}, j_{\mathrm{CV}}$ for the cylindrical CVs on the acceleration and the jerk. Notice we have to select $a_{\mathrm{CV}}, j_{\mathrm{CV}}$ satisfying Eq. (30) and (31).

The following shows how to constrain the derivatives of the trajectory within a convex polytope. The first derivative of a kth order B-spline $S_k^{(1)}$ is a $k-1$th order B-spline:

$$S_k^{(1)} = \frac{dS^k(s)}{dt} = \sum_{i=0}^{M-2} c_i^{(1)} N_{i+1}^{k-1}(s) \tag{33}$$

where $c_i^{(1)}$ is given as

$$c_i^{(1)} = \alpha \frac{k(c_{i+1}-c_i)}{s_{i+k+1}-s_{i+1}} \tag{34}$$

And the knot vector for $S_k^{(1)}$ is obtained by removing the first and last element in $\mathcal{K}$:

$$\begin{aligned} \mathcal{K}^{(1)} &= [s_0^{(1)}, s_1^{(1)} ..., s_{M+k-1}^{(1)}] \\ &= [\underbrace{0,\ldots,0}_{k \text{ times}}, 1, 2, \ldots, \underbrace{M-k,\ldots,M-k}_{k \text{ times}}]. \end{aligned} \tag{35}$$

Let the control point matrix of $S_k^{(1)}$ be $C^{(1)} = [c_0^{(1)}, c_1^{(1)}, \ldots, c_{(1)}^{(M-2)}]$, Eq. (34) can be expressed in a matrix form

$$\widehat{C}^{(1)} = \alpha \Gamma_1 \widehat{C}. \tag{36}$$

Following the same logic, for the nth order derivative $S_k^{(n)}$, its control points can be obtained as

$$\widehat{C}^{(n)} = \alpha^n \Gamma_n \widehat{C}. \tag{37}$$

Similar to Eq. (21), we also define the mapping matrix for the control points of $S_k^{(n)}$ as

$$c_i^{(n)} = \Lambda_i^{(n)} \widehat{C}^{(n)}. \tag{38}$$

Since $S_k^{(n)}$ is a $k-n$th order B-spline, by the analysis in Section IV-E, the sufficient condition for $S_k^{(n)}$ to stay in a close convex polytope whose interior is $\{p \in \mathcal{X} | A_n p \le b_n\}$, can be expressed as

$$\begin{aligned} A_n c_i^{(n)} =& A_n \Lambda_i^{(n)} \widehat{C}^{(n)} \\ =& \alpha^n A_n \Lambda_i^{(n)} \Gamma_n \widehat{C} < b_n,\ \forall i \in \{0,\ldots,M-n-1\} \end{aligned} \tag{39}$$

G. Boundary conditions

It is necessary to specify the initial and end state of the trajectory and its derivatives. We use $S_{\mathrm{ini}}, S_{\mathrm{end}}$ to denote the desired boundary condition of the trajectory S_k, and $S_{\mathrm{ini}}^{(n)}, S_{\mathrm{end}}^{(n)}$ to represent the desired boundary condition of the trajectory's nth derivative $S_k^{(n)}$. For a clamped B-spline its initial and end values equal to the first and last control points:

$$\begin{aligned} S_k(0) &= c_0 \\ S_k(\alpha T_{\mathrm{end}}) &= c_{M-1} \end{aligned} \tag{40}$$

The same goes for $S_k^{(n)}$ where

$$\begin{aligned} S_k^{(n)}(0) &= c_0^{(n)} \\ S_k^{(n)}(\alpha T_{\mathrm{end}}) &= c_{M-n-1}^{(n)} \end{aligned} \tag{41}$$

Combining Eq. (37), 21 and 38, the boundary conditions can be written as equality constraints over $\widehat{C}$ as

$$\begin{aligned} \Lambda_0 \widehat{C} &= S_{\mathrm{ini}} \\ \Lambda_{M-1} \widehat{C} &= S_{\mathrm{end}} \\ \alpha^n \Lambda_0^{(n)} \Gamma_n \widehat{C} &= S_{\mathrm{ini}}^{(n)} \\ \alpha^n \Lambda_{M-n-1}^{(n)} \Gamma_n \widehat{C} &= S_{\mathrm{end}}^{(n)} \end{aligned} \tag{42}$$

H. Qudratic programming problem

Since the cost functions Eq. (8), (10), (22) and (25) are all convex and quadratic whereas the constraints Eq. (29), (39) and (42) are all linear. The trajectory generation can be written as a QP in the form of

$$\begin{aligned} &\min_{\widehat{C}} \widehat{C}^{\intercal} G \widehat{C} + F \widehat{C} \\ &\text{s.t.} A_{\mathrm{eq}} \widehat{C} = b_{\mathrm{eq}} \\ &\quad A_{\mathrm{ieq}} \widehat{C} \le b_{\mathrm{ieq}} \end{aligned} \tag{43}$$

Similar to [5], if the problem does not require the inequality constraints in Eq. (43), and its equality constraints only contain boundary conditions, a closed form solution is possible by solving the boundary condition independently. From the construction of A_{eq} (Eq. (42)), we notice that all elements between its $k+1$th column until $M-k$th column are all zero. Therefore, we make a new matrix

$$\begin{aligned} A_{\mathrm{F}} = [&A_{\mathrm{eq}_{(*,0)}}, A_{\mathrm{eq}_{(*,1)}}, \ldots, A_{\mathrm{eq}_{(*,k)}}, \\ &A_{\mathrm{eq}_{(*,M-k+1)}}, \ldots, A_{\mathrm{eq}_{(*,M)}}] \end{aligned} \tag{44}$$

and re-arrange the control point matrix C into two parts

$$C_F = [c_0, c_1, \ldots, c_{k-1}, c_{M-k}, \ldots, c_{M-1}]$$
$$C_M = [c_k, c_{k+1}, \ldots, c_{M-k-1}] \tag{45}$$

where C_F is determined by the boundary conditions as

$$C_F = A_F^{-1} b_{\text{eq}}. \tag{46}$$

Let $G_\Phi = \Phi^\intercal G \Phi$, $F_\Phi = F\Phi$, where Φ is a mapping matrix

$$\widehat{C} = \Phi \begin{bmatrix} \widehat{C}_F \\ \widehat{C}_M \end{bmatrix}, \tag{47}$$

then the cost function in Eq. (43) can be rewritten as

$$[\widehat{C}_F^\intercal \ \widehat{C}_M^\intercal] G_\Phi \begin{bmatrix} \widehat{C}_F \\ \widehat{C}_M \end{bmatrix} + F_\Phi \begin{bmatrix} \widehat{C}_F \\ \widehat{C}_M \end{bmatrix} \tag{48}$$

And by splitting G_Φ and F_Φ according to the dimension of $\widehat{C}_F$ and $\widehat{C}_M$, the cost function now equlas

$$[\widehat{C}_F^\intercal \quad \widehat{C}_M^\intercal] \begin{bmatrix} G_\Phi^{FF} & G_\Phi^{FM} \\ G_\Phi^{MF} & G_\Phi^{MM} \end{bmatrix} \begin{bmatrix} \widehat{C}_F \\ \widehat{C}_M \end{bmatrix} + [F_\Phi^F \quad F_\Phi^M] \begin{bmatrix} \widehat{C}_F \\ \widehat{C}_M \end{bmatrix} \tag{49}$$

where the minimum is achieved at

$$\widehat{C}_M = (G_\Phi^{MM})^{-1} (\widehat{C}_F^\intercal G_\Phi^{FM} + \widehat{C}_F^\intercal (G_\Phi^{MF})^\intercal + F_\Phi^M) \tag{50}$$

V. Flight Experiments and Analysis

The presented method is numerically robust and efficient. Its capability are demonstrated through example applications and comparisons with a 4th order B-spline ($k = 4$) as a running example. For the interpolation method using polynomial splines [4], [5], we adopt an open source implementation from [18] which uses the 7th order polynomial. All comparison and evaluation is done in Matlab with the only exception in calculating H (Eq. (10)) where an open source C program is linked against Matlab through Mex. The video of flight experiments is at: `http://uav.ece.nus.edu.sg/videos_files/2018/bspline.mp4`.

A. Fitting sketching inputs

With our method, it is possible to directly fit user sketching inputs. Moreover, a sketching input is desirable when the required flying pattern is complex and contains subtle details, such as the one for filming and light painting. The raw sketching input normally consists of large amount of densely located waypoints and is difficult for previous interpolation based methods [4], [5] to handle. The major challenges are:

1) Time-point estimation: The time-point of each waypoint is usually estimated heuristically for efficiency. However, inaccurate time-point would introduce excursions and wobbling in the trajectory [12]. While in [4], [5], numerical methods is proposed for time-point optimization, its efficiency and stability remains an open challenge [12]. On the other hand, our approximation approach could handle crudely estimated time-points by assigning smaller weights. In Fig. 2, the approximation and interpolation approach is compared. The former gives a trajectory with a smoother velocity profile and a better fitting quality.

2) Numerical stability: The interpolation method becomes less numerically stable when facing large amounts of densely located waypoints. This is because the resulting QP of the interpolation approach have an positive-semi definite hessian matrix. For our formulation in Eq. (6), though each individual term in the cost function is positive-semi definite, their sum is usually positive definite except some rare cases. Fig. 3 shows the result of fitting and tracking user sketching data which has 1764 waypoints using the proposed method while the interpolation method diverges when given the same set of time-point allocation. However, if there is only one non-zero term in the cost function, the final hessian will still be positive-semi definite.

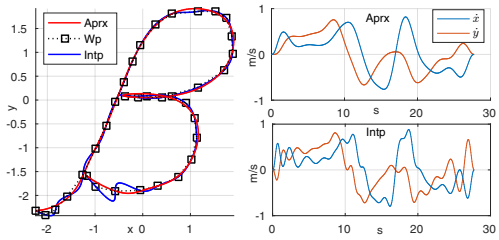


Fig. 2. Fitting of a user written character β. The approximation method (Aprx) utilizes the proposed B-spline. The interpolation method (Intp) utilizes a 7-order polynomial spline. The time-points allocation are the same for both methods.

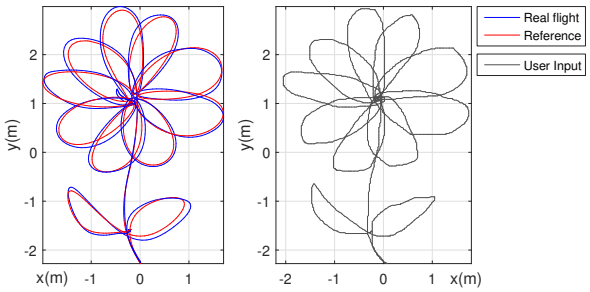


Fig. 3. Densely fitting of user sketching inputs. The blue curve shows the real vehicle tracking performance.

B. Approximating line-segments

The waypoint interpolation is prone to poorly chosen time-points which leads to large excursions [12]. To suppress the excursion, [5] chooses to insert intermediate waypoints along the underlying straight lines. However, it might lead to an increase in the snap cost. An novel ad hoc method is proposed to suppress the excursion. It bends at each waypoint instead of passing through them to avoid aggressive maneuver. Firstly, the associated time-points is estimated. In Fig. 4, let each waypoint Wp_i has time-point t_i where

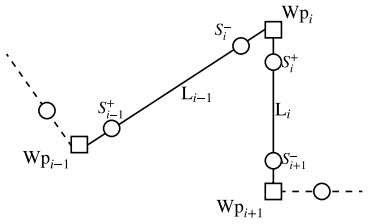


Fig. 4. Problem construction of the proposed method

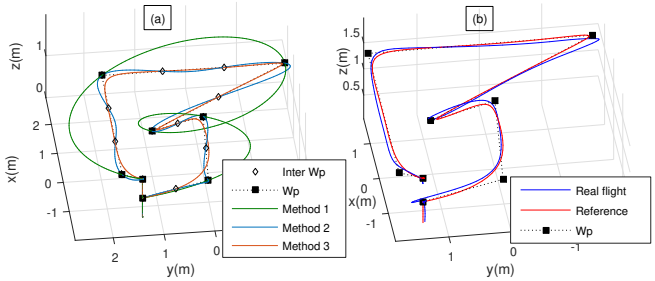


Fig. 5. Comparison for fitting line segments. The time-point allocation for the original waypoints are the same for each method.

$i \in \{0, 1, \ldots, N_{\mathrm{W}} - 1\}$. Similar to [9], $\Delta t_i = t_{i+1} - t_i$ is calculated by solving an acceleration limited time optimal trajectory along the line L_i and stops at Wp_i and Wp_{i+1}. Then, two intermediate waypoints S_i^+ and S_{i+1}^- are selected from the time optimal trajectory at moments $t_i + \kappa\Delta t_i$ and $t_i + (1-\kappa)\Delta t_i$ where κ is a design variable. These moments also serve as the time-points for S_i^+ and S_{i+1}^-. To approximate the line-segments, we choose to

- minimize the distance to $\mathrm{Wp}_i, S_i^-, S_i^+$ at their corresponding time-points for $i \in \{0, 1, \ldots, N_{\mathrm{W}} - 1\}$.
- minimize the deviation from L_i over time interval $[t_i + \kappa\Delta t_i,\ t_i + (1-\kappa)\Delta t_i)$ for $i \in \{0, 1, \ldots, N_{\mathrm{W}} - 2\}$.
- penalize also the lower derivatives such as velocity and acceleration. In particular, $\sigma = [0.5, 0.707, 0, 1]^\intercal$ (Eq. (7)).

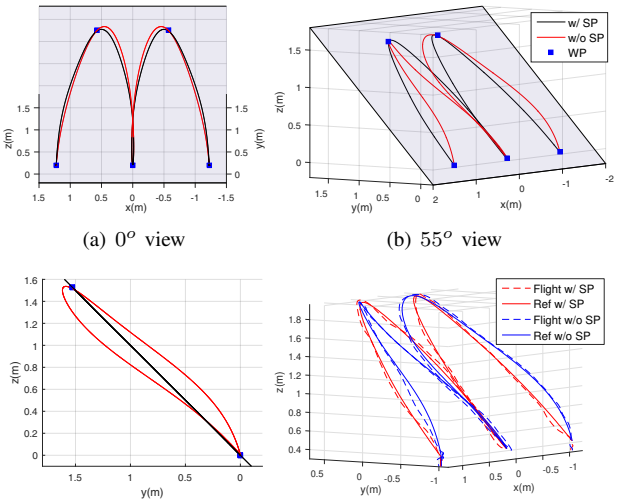
Fig. 5 shows an comparison between 3 methods. Method 1 interpolates the original waypoints only. Method 2 insert intermediate waypoints and interpolate all waypoints. For a line-segment with end-points Wp_i, Wp_{i+1}, the intermediate waypoints are sampled uniformly in between and their time-points is calculated by assuming a constant speed $\left\|\mathrm{Wp}_i - \mathrm{Wp}_{i+1}\right\| / \Delta t_i$. Both method 1 & 2 use the 7th order polynomial spline. Finally, method 3 is the proposed method. Method 2 and 3 perform similarly in terms of suppressing the excursion. The average distances to the line-segments are 0.05m and 0.02m for method 2 and 3 respectively. However, method 3 produces an less aggressive trajectory with a snap cost of 348 while the method 2 gives a much higher snap cost of 6923.

Tab. I compares different methods on 100 sets of randomly generated trails. Each trail contains 5 waypoints chosen in a $50\,\mathrm{m} \times 50\,\mathrm{m} \times 20\,\mathrm{m}$ volume. Method A is the proposed ad hoc method with $\omega = [0.5, 0.5, 0]^\intercal$ (Eq. (6)), $\sigma = [0.5, 0.707, 0, 1]^\intercal$ and $M = 100$. Method B draws intermediate waypoints and approximates them using B-spline with $\omega = [1, 0, 0]^\intercal$, $\sigma = [0.5, 0.707, 0, 1]^\intercal$ and $M = 100$. Method C is similar to method B but with $\sigma = [0, 0, 0, 1]^\intercal$. Method D also draws intermediate waypoints but interpolates them with the 7th order polynomial spline and $\sigma = [0, 0, 0, 1]^\intercal$. All methods have the same time-point allocation for the original 5 wayponints. Moreover, for methods B, C and D, their intermediate waypoints and corresponding time-points are the same. The methods are compared in terms of snap cost, average distance to line segments (Avg.dist), maximum distance to line segments (Max.dist) and computational time (Com.t). From Tab. I, it is noticed that by approximating instead of interpolating the waypoints, the snap cost and fitting quality are both improved. By penalizing the lower derivatives, the overshoot is reduced and hence the fitting quality is improved. Finally, the proposed method further reduces the snap cost by fitting the underlying straight lines directly.

TABLE I

COMPARISON IN FITTING LINE-SEGMENTS

Method	Snap cost	Avg.dist(m)	Max.dist(m)	Com.t(ms)
A	15.6	0.08	1.0	7.8
B	41.5	0.08	1.0	11.4
C	41.9	0.3	2.1	6.2
D	84544.2	0.63	4.9	23.2



(a) 0^o view (b) 55^o view

(c) 90^o view (d) Real flight

Fig. 6. Comparison between trajectory with surface penalty (w/SP) and without surface penalty (w/o SP).

C. Generating trajectories on desired planes

It is sometimes desirable to force the trajectory to stay on certain two dimensional planes. For example, in a light paint event, it helps to prevent the designed shape from distorting and avoid collision between vehicles from different planes. The results in Sec.IV-D can be applied directly to achieve the desired behavior. In Fig. 6, the trajectory is designed to write the letter M for a light paint performance. The waypoints that define the letter reside on the same surface, namely the painting surface. However, if we only interpolate these waypoints, the different velocity and acceleration limits on the horizontal and vertical axes cause an asynchronous movement perpendicular to the painting surface which distorts the letter from 55^o side view (Fig. 6(b)). On the other hand, with our formulation, we can easily penalize the distance to the painting surface and force the trajectory to stay on it. In real life application, the distortion is more obvious when the horizontal and vertical speed limits varies significantly. The phenomenon is demonstrated in an indoor environment, the velocity and acceleration limits are chosen as $|\dot{x}| < 2$, $|\dot{y}| < 2$, $|\dot{z}| < 0.2$, $|\ddot{x}| < 2$, $|\ddot{y}| < 2$, $|\ddot{z}| < 0.2$.

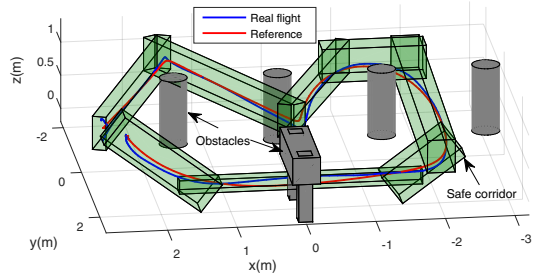
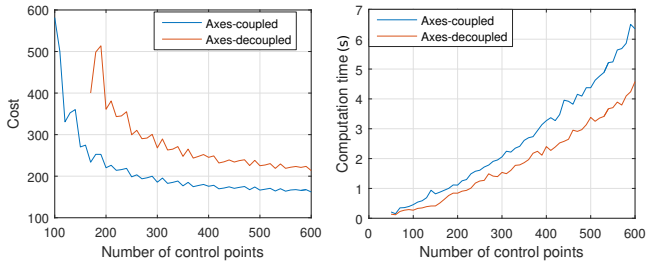


Fig. 7. Safe corridor



(a) Cost (b) Computing time

Fig. 8. Comparison between axes-coupled and decoupled methods

D. Safety and feasibility

In our method, the safety and feasibility of the trajectory is guaranteed by constraining the trajectory and its derivatives in safe regions consists of convex polytopes. In Fig. 7, a safe corridor consists of 7 oriented rectangles with the generated trajectory is shown. The safe corridor is designed by human beings, but it can also be generated automatically in realtime using the algorithm in [2]. The safe zone constraint is handled by applying the results in Sec.IV-E. The time interval for each corridor is determined by the method in Sec.V-B as the minimum time to cover the length of the corridor. The feasibility condition is specified by cylindrical CVs on the trajectory's velocity, acceleration and jerk with $v_{\mathrm{SCV}} = [3.1, -0.55, 2.2]$, $a_{\mathrm{CV}} = [2.8, -0.5, 2]$, $j_{\mathrm{CV}} = [7.1, -5, 5]$. To obtain linear inequalities, the axes-decoupled method constructs the largest axis-aligned cuboid within each cylinder while the axes-coupled method constructs the largest hexagonal prism. Their performance in the optimization problem in terms of cost value and computational time is given in Fig. 8. The axes-coupled inequality constraints gives a lower cost with slightly longer computational time. It gives feasible solutions from 100 control points onward whereas the axes-decoupled method only gives feasible solutions from 170 control points onward.

VI. Conclusion

In this paper, we present an optimal constrained trajectory generation method for quadrotors using clamped uniform B-splines. It systematically addresses the issue of interval-wise and axes-coupled constraint, both on the trajectory itself and its derivatives. These constraints can be made of arbitrarily shaped convex polytopes and satisfied on the entire trajectory. Besides, we also propose a method to directly penalize the trajectory's deviation from desired lines and planes without drawing intermediate waypoints. A formulation to suppress the excursion between waypoints is presented. All the mentioned problems can be combined and solved efficiently in a single QP. Moreover, a closed-form solution is presented for the case without inequality constraints. We demonstrate its numerical stability by examples of fitting thousands of waypoints. The proposed approach has been tested with various real flight experiments.

References

[1] C. Gebhardt, B. Hepp, T. Nägeli, S. Stevšić, and O. Hilliges, "Airways: Optimization-based planning of quadrotor trajectories according to high-level user goals," in *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*, 2016, pp. 2508–2519.

[2] S. Liu, M. Watterson, K. Mohta, K. Sun, S. Bhattacharya, C. J. Taylor, and V. Kumar, "Planning dynamically feasible trajectories for quadrotors using safe flight corridors in 3-d complex environments," *IEEE Robotics and Automation Letters*, vol. 2, no. 3, pp. 1688–1695, July 2017.

[3] M. Hehn and R. DAndrea, "Real-time trajectory generation for quadrocopters," *IEEE Transactions on Robotics*, vol. 31, no. 4, pp. 877–892, Aug 2015.

[4] D. Mellinger and V. Kumar, "Minimum snap trajectory generation and control for quadrotors," in *2011 IEEE International Conference on Robotics and Automation*, May 2011, pp. 2520–2525.

[5] C. Richter, A. Bry, and N. Roy, "Polynomial trajectory planning for quadrotor flight," in *RSS Workshop on Resource-Efficient Integratiton of Perception, Control and Navigation for MAVs*, 2013.

[6] J. Chen, T. Liu, and S. Shen, "Online generation of collision-free trajectories for quadrotor flight in unknown cluttered environments," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*, May 2016, pp. 1476–1483.

[7] S. Tang and V. Kumar, "Safe and complete trajectory generation for robot teams with higher-order dynamics," in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Oct 2016, pp. 1894–1901.

[8] J. A. Preiss, W. Honig, N. Ayanian, and G. S. Sukhatme, "Downwash-aware trajectory planning for quadrotor swarms," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Sep 2017, pp. 250–257.

[9] S. Lai, K. Wang, and B. M. Chen, "Dynamically feasible trajectory generation method for quadrotor unmanned vehicles with state constraints," in *2017 36th Chinese Control Conference (CCC)*, July 2017, pp. 6252–6257.

[10] H. Kano, H. Fujioka, and C. F. Martin, "Optimal smoothing and interpolating splines with constraints," *Applied Mathematics and Computation*, vol. 218, no. 5, pp. 1831 – 1844, 2011.

[11] H. Kano and H. Fujioka, "Velocity and acceleration constrained trajectory planning by smoothing splines," in *2017 IEEE 26th International Symposium on Industrial Electronics (ISIE)*, June 2017, pp. 1167–1172.

[12] S. Tang and V. Kumar, "Autonomous flying," *Annual Review of Control, Robotics, and Autonomous Systems*, vol. 1, no. 1, 2018.

[13] M. W. Mueller, M. Hehn, and R. D'Andrea, "A computationally efficient algorithm for state-to-state quadrocopter trajectory generation and feasibility verification," in *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Nov 2013, pp. 3480–3486.

[14] M. Hehn, R. Ritz, and R. D'Andrea, "Performance benchmarking of quadrotor systems using time-optimal control," *Autonomous Robots*, vol. 33, no. 1, pp. 69–88, Aug 2012.

[15] T. Kröger and F. M. Wahl, "Online trajectory generation: Basic concepts for instantaneous reactions to unforeseen events," *IEEE Transactions on Robotics*, vol. 26, no. 1, pp. 94–111, Feb 2010.

[16] S. Lai, K. Wang, H. Qin, J. Q. Cui, and B. M. Chen, "A robust online path planning approach in cluttered environments for micro rotorcraft drones," *Control Theory and Technology*, vol. 14, no. 1, pp. 83–96, Feb 2016.

[17] B. T. Lopez and J. P. How, "Aggressive 3-d collision avoidance for high-speed navigation," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, May 2017, pp. 5759–5765.

[18] Y. Lu, M. Cai, W. Ling, and X. Zhou. Quadrotor control, path planning and trajectory optimization. [Online]. Available: https://github.com/stormmax/quadrotor