An Efficient Approach of Time-optimal Trajectory Generation for the Fully Autonomous Navigation of the Quadrotor

Wei Dong, Ye Ding, Jie Huang, Xiangyang Zhu, Han Ding,
State Key Laboratory of Mechanical System and Vibration,School of Mechanical Engineering
Shanghai Jiao Tong University,Shanghai, 200240, China
Email: chengquess@sjtu.edu.cn, y.ding@sjtu.edu.cn, thk2dth@sjtu.edu.cn, mexyzhu@sjtu.edu.cn, hding@sjtu.edu.cn

In this work, a time-optimal trajectory generation approach is proposed for the multiple way-point navigation of the quadrotor based on the non-uniform rational B-spline (NURBS) curve and linear programming. To facilitate the development of the trajectory generation approach, the dynamic model of the quadrotor is formulated. Then the geometric trajectory regarding multiple way-point navigation is constructed based on the NURBS curve. With the constructed geometric trajectory, a time-optimal interpolation problem is imposed considering the velocity, acceleration and jerk constraints. This optimization problem is solved in two steps. At the first step, a preliminary result is obtained by solving a linear programming problem without jerk constraints. Then by introducing the properly relaxed jerk constraints, an additional linear programming problem is formulated based on the preliminarily obtained result and the time-optimal problem can be fully solved in this way. Subsequently, a nonlinear trajectory tracking controller is developed to track the generated trajectory. The feasibilities of the proposed trajectory generation approach as well as the tracking controller are verified through both simulations and real-time experiments. In the simulations, the advantages of the proposed approach compared to existing trajectory generation methods are well demonstrated with the efficient computation and significantly reduced flight time cost. The experimental results also show that with velocity, acceleration, and jerk constraints: $V_{\max} = 1\ m/s$, $A_{\max} = 2\ m/s^2$, and $J_{\max} = 5\ m/s^3$, the proposed approach can generate trajectory with the smooth acceleration profile and moderate velocity $V \approx 1$ m/s. In such a case, the trajectory tracking controller can closely track the reference trajectory with cross-tracking error less than 0.05 m. All of these results show the superiorities of the proposed time-optimal trajectory generation approach for the fully autonomous navigation of the quadrotor.

1 Introduction

Owing to the achievement of the energy, electronics, and communication technologies, there is a boom in the development of flying robots, such as rotating-wing robots and flapping-wing robots [1]. Nowadays, the activities of these robots can significantly extend into three-dimensional space. Among all of the flying robots, the performance of the propeller-based flying robots, especially the quadrotor, are well recognized [1, 2]. The quadrotor has already been extensively applied for photography [3] and infrastructure inspection [4, 5], and more agile and intellectual capabilities of the quadrotor, such as avian-inspired manipulation [6, 7], ball juggling [8–10], flying pendulum [11], cooperation [12], and minimum snap tracking [13], were demonstrated by researchers recently [14]. In this process, the quadrotor dynamics [15, 16], flight control [17–19], and trajectory generation [9, 20, 21] are extensively investigated.

In particular, dynamically consistent trajectories are necessary for the quadrotor to fulfill corresponding missions autonomously. Due to the nonlinear and under-actuated nature of the quadrotor, the trajectory generation might be complicated and time consuming [22, 23]. To address this issue, various algorithms were proposed considering the trade-offs between the computational efficiency and optimality of the trajectory. These algorithms are generally developed in two different approaches [22, 24]. In the first approach, the trajectory is obtained by directly solving the optimization problem over the objective function, such as minimum time [22] or minimum snap [13], under geometric or input constraints. In the second approach, a geometric trajectory is preliminarily constructed by splines[25], polynomials [26], or trigonometric functions [27], and the constructed trajectory is then parameterized in time domain to guarantee the dynamic consistency.

Broadly speaking, the optimal trajectory addressed as two-point boundary value problems have been explicitly solved [22, 25]. However, it is difficult or time-consuming to obtain the globally optimized trajectory, especially in the case of multiple way-points with geometric or input constraints [26], in the real-time flights. This is because on one hand the continuities in position, velocity and acceleration should be guaranteed in all of the way-points, and on the other hand, the objective functions, e.g., the minimum jerk, are commonly imposed associating with optimal time arrangement [13], which is nonlinear and also time consuming. For this reason, although in [26] a minimum snap trajectory generation approach was well proposed for structured environment, in autonomous navigation under unknown and cluttered environment, the reference trajectories were still commonly generated with straight line solution using the RRT* [28–30], which is obviously not dynamically consistent. To tackle this issue, researchers have to either investigate computationally efficient trajectory generation methods, properly re-formulate the optimal problem or relax the constraints regarding the specific missions. For example, in [31] the authors properly re-formulated the optimal problem based on [13, 32], and established a more efficient real-time trajectory generation approach. As demonstrated in [31, 32], it is worthing to properly sacrifice the optimality of the trajectory, in order to enhance the computational efficiency. This is because in real-time flight, it is not always possible for the quadrotor to hover for tens of seconds to calculate the global optimal trajectory of relative short distance, which implies that the computational efficiency is often important than the optimality for the quadrotor to pass through predefined waypoints following a feasible trajectory [21,33].

Actually, in the autonomous navigation such as demonstrated in [31], the navigation way-points are always firstly searched within the occupancy map. In addition, in civil applications, such as delivery [34] and bridge inspection [35], the geometric trajectories are also predefined regarding the specific tasks. In such a case, it is not necessary and even impossible in real-time flight to pursue global optimal trajectory with objective in both minimum time and minimum jerk or snap, etc [33]. Instead, it is might be more nature and practice to first construct dynamically feasible geometric trajectory and then carry out time-optimal interpolation, which would effectively enhance the computational efficiency. For this reason, researchers have started to develop trajectory generation strategies based on the collocation approach, and several effective methodologies have been proposed in the robot community recently [36–38]. Among these methods, the spline-based approach provides promising solution for the real-time trajectory generation owing to its mathematical conciseness. However, to the best knowledge of the authors, the existed approaches commonly based on nonlinear programming, the computational efficiency of which could be further enhanced. In such a case, it might be desired if more computationally effective optimization method could be proposed. For example, if one can properly convert the nonlinear programming into linear programming, both the computational efficiency and accuracy could be extensively improved in the real-time applications.

In view of the state-of-the-art, this work is motivated to propose a time-optimal trajectory generation approach for multiple way-point navigation of the quadrotor based on the non-uniform rational B-spline (NURBS) curve and linear programming. To facilitate the development of the trajectory generation approach, the dynamic model of the quadrotor is first formulated. Then the geometric trajectory regarding multiple way-point navigation is constructed based on the NURBS curve. With the constructed geometric trajectory, a time-optimal problem is imposed considering the velocity, acceleration and jerk constraints. This optimization problem is solved in two steps. At the first step, a preliminary result is obtained by solving a linear programming problem without jerk constraints. Subsequently, substituting the obtained result and properly relaxing the jerk constraints, an addition linear programming problem is introduced and the time-optimal problem can be fully solved. A nonlinear trajectory tracking controller is then developed, and the feasibilities of the proposed trajectory generation approach as well as the tracking controller are verified through both simulations and real-time experiments.

The distinctive features of this paper are as follows. Firstly, the spline-based geometric trajectory generation approach for the multiple way-point navigation of the quadrotor is proposed in an analytic way. Secondly, the time-optimal trajectory sequence is converted into two linear programming problems with velocity, acceleration and jerk constraints, which is very important to improve the computational efficiency and implementing convenience in practice. Compared to the existed nonlinear method, the computational efficiency and reliability of the trajectory generation approach are effectively enhanced. Finally, a nonlinear trajectory tracking controller developed in a compact structure is proposed to closely track the generated trajectory.

The reminder of this paper is organized as follows. The quadrotor model is presented in Section 2, and the trajectory generation as well as the time-optimal interpolation is provided in Section 3. The trajectory tracking controller is developed in Section 4. Subsequently, the numeric simulations and real-time experiments are conducted in Section 5. The conclusion is provided in Section 6.

2 Dynamic Model

To facilitate the trajectory generation and controller design, the dynamic model of the quadrotor is first formulated regarding the existed works in this section. Based on this model, the differential flatness property of the quadrotor is also briefly introduced for high performance trajectory tracking control.

2.1 Rigid Body Dynamics

The coordinates and free body diagram of the quadrotor are shown in Fig. 1. According to this diagram, four control inputs can be defined as

$$\begin{aligned} G_1 &= F_1 + F_2 + F_3 + F_4, & G_2 &= (F_2 - F_4)L, \\ G_4 &= M_1 - M_2 + M_3 - M_4, & G_3 &= (F_3 - F_1)L. \end{aligned} \tag{1}$$

where L is the length from the rotor to the center of the mass of the quadrotor, and F_i and M_i are the thrust and torque generated by rotor i ($i \in \{1,\ 2,\ 3,\ 4\}$).

In this way, the dynamic model of the quadrotor can be

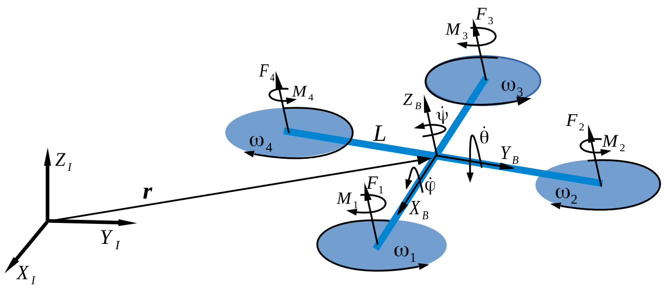


Fig. 1. Free body diagram

expressed as

$$\begin{cases} m\ddot{\mathbf{r}} = \mathbf{R}_I^{\mathcal{B}}(G_1\mathbf{Z}_B) - mg\mathbf{Z}_I \\ \mathbf{I}\ddot{\mathbf{q}} = [G_2, G_3, G_4]^T - S(\mathbf{Dq})\mathbf{I}(\mathbf{Dq}) \end{cases} \tag{2}$$

where $\mathbf{r} = [x,y,z]^T$ is the position of the center of mass in the inertial coordinates I, $\mathbf{q} = [\phi,\theta,\psi]^T$ is the attitude, m and $\mathbf{I}$ are the mass and moment of inertia of the quadrotor, g is the gravity constant, $\mathbf{R}_I^{\mathcal{B}}$ is the transform matrix between the inertial frame I and the body-fixed frame $\mathcal{B}$ as defined in Fig. 1, $\mathbf{D}$ represents the affine from the attitude angles to the angular velocities [22] and $S(\cdot)$ is the screw matrix representation of the corresponding vector [39].

In view of Eq. (2), equations governing dynamics of the quadrotor with respect to the inertial coordinates can be expanded as [40, 41]

$$\begin{cases} \ddot{x} = \frac{G_1}{m}(\cos\phi\sin\theta\cos\psi + \sin\phi\sin\psi) \\ \ddot{y} = \frac{G_1}{m}(\cos\phi\sin\theta\sin\psi - \sin\phi\cos\psi) \\ \ddot{z} = \frac{G_1}{m}\cos\phi\cos\theta - g \\ \ddot{\phi} = \frac{G_2}{I_{xx}} + \dot{\theta}\dot{\psi}(\frac{I_{yy}-I_{zz}}{I_{xx}}) \\ \ddot{\theta} = \frac{G_3}{I_{yy}} + \dot{\phi}\dot{\psi}(\frac{I_{zz}-I_{xx}}{I_{yy}}) \\ \ddot{\psi} = \frac{G_4}{I_{zz}} + \dot{\phi}\dot{\theta}(\frac{I_{xx}-I_{yy}}{I_{zz}}) \end{cases} \tag{3}$$

where I_{xx}, I_{yy}, and I_{zz} are the moments of inertia of the quadrotor in the x, y, and z axis, respectively.

In this work, the output space is selected as $\boldsymbol{\zeta} = \mathbf{r}\times\dot{\mathbf{r}}\times\ddot{\mathbf{r}}$. As the quadrotor is a differential flatness system [13, 26], there exists a mapping $\mathbf{r} \mapsto \{\mathbf{q}, G_1\} : [\phi,\theta,G_1]^T = \lambda_p(\mathbf{r})$. In this way, once the desired trajectory $\mathbf{r}^*$ is designed, the state trajectory, say $\boldsymbol{\xi}^*$, of the quadrotor can also be fully determined. In particular, the attitude and overall thrust can be expressed as [26]

$$\begin{array}{ll} \phi = \arcsin(\frac{-m\ddot{y}}{G_1}) & \theta = \arctan(\frac{\ddot{x}}{\ddot{z}+g}) \\ G_1 = m\sqrt{\ddot{x}^2 + \ddot{y}^2 + (\ddot{z}+g)^2} & \end{array} \tag{4}$$

When $\psi = 0$. With this conception, an efficient trajectory tracking controller will be developed to closely track the generated reference trajectory in Section 4.

3 Trajectory generation and time-optimal interpolation

In this section, a trajectory generation approach is proposed based on the NURBS curve, and a time-optimal interpolation method is proposed to generate real-time trajectory reference by constructing two linear programming problems with constraints on velocity, acceleration and jerk [42, 43].

3.1 Trajectory generation based on the NURBS curve

The NURBS curve is a parametric curve. This kind of curve is a function $\mathbf{C}(u)$ that returns a point $\mathbf{C}(u_i)$ on the curve for a particular value of the parameter u. It should be noted that the parameter u is not directly related to the time variable t. The NURBS curve itself is defined by knot vector $\mathbf{U}$, which is a sequence of parameter values that determines where and how the control points affect the NURBS curve, control points $\{\mathbf{P}_i\}$, weight coefficient w_i and the order of the curve k [44, 45].

In real-time flights, it is often required to first calculate the desired control points $\mathbf{P}_i$ based on the desired navigation way-points $\{\mathbf{Q}_i^*\}$, or namely data points. The control points can be obtained as follows. Denoting the way-points passed through by the NURBS curve generated with $\mathbf{P}_i$ and u_i are $\{\mathbf{Q}_i\}$, the following relationship is established [44, 45]

$$\underbrace{\begin{bmatrix} \mathbf{Q}_0 \\ \mathbf{Q}_1 \\ \vdots \\ \mathbf{Q}_n \end{bmatrix}}_{\mathbf{Q}} = \underbrace{\begin{bmatrix} N_{0,k}(u_0) & N_{1,k}(u_0) & \cdots & N_{n,k}(u_0) \\ N_{0,k}(u_1) & N_{1,k}(u_1) & \cdots & N_{n,k}(u_1) \\ \vdots & \vdots & \ddots & \vdots \\ N_{0,k}(u_n) & N_{1,k}(u_n) & \cdots & N_{n,k}(u_n) \end{bmatrix}}_{\mathbf{\Phi}} \underbrace{\begin{bmatrix} \mathbf{P}_0 \\ \mathbf{P}_1 \\ \vdots \\ \mathbf{P}_n \end{bmatrix}}_{\mathbf{P}} \tag{5}$$

where $N_{i,k}$ is the NURBS basis function [44, 45].

As the desired navigation way-points are $\{\mathbf{Q}_i^*\}$, to minimize the distance between the desired way-points $\{\mathbf{Q}_i^*\}$ and the generated NURBS curve, the optimal problem can be imposed as

$$\min J = \frac{1}{2}(\mathbf{Q}^* - \mathbf{Q})^T(\mathbf{Q}^* - \mathbf{Q}) = \frac{1}{2}(\mathbf{Q}^* - \mathbf{\Phi}\mathbf{P})^T(\mathbf{Q}^* - \mathbf{\Phi}\mathbf{P}) \tag{6}$$

When the number of the way-points is small, one can adopt the least square method to estimate the optimal solution of Eq. (6) as follows

$$\mathbf{P} = (\mathbf{\Phi}^T\mathbf{\Phi})^{-1}\mathbf{\Phi}^T\mathbf{Q} \tag{7}$$

where $\mathbf{\Phi}^T\mathbf{\Phi}$ is mostly invertible, since $\mathbf{\Phi}$ is a quasi-diagonal matrix, and all the operation is carried out with float point data, which means it is commonly not possible for a single eigenvalue to be exactly zero. To safely avoid the singular case of the matrix $\mathbf{\Phi}^T\mathbf{\Phi}$, one can further adopt the Levenberg-Marquardt Method, also known as the damped least square method. In the simulation and experiments conducted in this paper, the matrices $\mathbf{\Phi}$s are all found to be full rank, therefore, Eq. (7) is directly utilized to solve the control points.

3.2 Interpolation

In Section 3.1, the geometric trajectory is established in the form of a function with respect to the parameter u, which is not directly related to the time t. In order to properly conduct the temporal planning, this section will first derive the implicit time function of the variant u.

For a given curve $\mathbf{C}(u)$, its velocity can be expressed as [42,43].

$$V(u_i) = \left\| \frac{\mathrm{d}\mathbf{C}(u)}{\mathrm{d}t} \right\|_{u=u_i} = \left\| \frac{\mathrm{d}\mathbf{C}(u)}{\mathrm{d}u} \right\|_{u=u_i} \cdot \left. \frac{\mathrm{d}u}{\mathrm{d}t} \right|_{t=t_i} \tag{8}$$

where $\mathrm{d}u/\mathrm{d}t > 0$ since u is assumed to increase with time in the temporal planning. If and only if all components of $\mathbf{C}(u)$ simultaneously achieve their local minima or maxima, $\|\mathrm{d}\mathbf{C}(u)/\mathrm{d}u\|$ equals to zero. This is unlikely to happen, especially in the numeric interpolation with float point operation. In the case $\left\|\mathrm{d}\mathbf{C}(u^{\#})/\mathrm{d}u\right\| = 0$ when $u = u^{\#}$, one can also choose to slightly vary the parameter variable u for a small value δu and re-calculate the derivative at the nearby point $u = u^{\#} + \delta u$ in the following interpolation, and thereby avoid such singular point.

In this way, (8) is equivalent to the following relationship

$$\dot{u}|_{t=t_i} = \left. \frac{\mathrm{d}u}{\mathrm{d}t} \right|_{t=t_i} = \frac{V(u_i)}{\left\| \frac{\mathrm{d}\mathbf{C}(u)}{\mathrm{d}u} \right\|_{u=u_i}} \tag{9}$$

Taking the derivative of (9), the second derivative of u can be calculated as [43].

$$\ddot{u}|_{t=t_i} = \frac{\mathrm{d}}{\mathrm{d}t} \left(\frac{V(u_i)}{\left\| \frac{\mathrm{d}\mathbf{C}(u)}{\mathrm{d}u} \right\|_{u=u_i}} \right) = \frac{-V(u_i)^2}{\left\| \frac{\mathrm{d}\mathbf{C}(u)}{\mathrm{d}u} \right\|^3_{u=u_i}} \cdot \frac{\left\| \frac{\mathrm{d}\mathbf{C}(u)}{\mathrm{d}u} \right\|_{u=u_i}}{\mathrm{d}u} \tag{10}$$

Substituting (9) into (10), one can obtain

$$\ddot{u}|_{t=t_i} = -\frac{V^2(u_i)\left(\frac{\mathrm{d}\mathbf{C}(u)}{\mathrm{d}u} \cdot \frac{\mathrm{d}^2\mathbf{C}(u)}{\mathrm{d}u^2} \right)}{\left\| \frac{\mathrm{d}\mathbf{C}(u)}{\mathrm{d}u} \right\|^4_{u=u_i}} \tag{11}$$

where the operation $\cdot$ is defined as the inner product, i.e., $\frac{\mathrm{d}\mathbf{C}(u)}{\mathrm{d}u} \cdot \frac{\mathrm{d}^2\mathbf{C}(u)}{\mathrm{d}u^2} = (\frac{\mathrm{d}\mathbf{C}(u)}{\mathrm{d}u})^T \frac{\mathrm{d}^2\mathbf{C}(u)}{\mathrm{d}u^2}$.

The discrete form of u_{i+1} can be obtained by taking Taylor's series as follows

$$u_{i+1} = u_i + \dot{u}(t)T_s|_{t=t_i} + \ddot{u}(t)\left.\frac{T_s^2}{2}\right|_{t=t_i} + O(t^3) \tag{12}$$

where T_s is the step length of the interpolation.

According to (9) and (11), one can obtain

$$u_{i+1} = u_i + \frac{V(u_i)T_s}{\left\| \frac{\mathrm{d}\mathbf{C}(u)}{\mathrm{d}u} \right\|_{u=u_i}} - \frac{V^2(u_i)T_s^2\left(\frac{\mathrm{d}\mathbf{C}(u)}{\mathrm{d}u} \cdot \frac{\mathrm{d}^2\mathbf{C}(u)}{\mathrm{d}u^2} \right)}{2\left\| \frac{\mathrm{d}\mathbf{C}(u)}{\mathrm{d}u} \right\|^4_{u=u_i}} \tag{13}$$

where $V(u_i)$ is the flight speed, which will be planned as follows.

3.3 Optimal velocity planning

For a given parameterized curve, it is desired to fulfill the flight mission with minimum time. This objective can be expressed as the following optimum problem [42,43]

$$\min T_{\Sigma} = \min_{\dot{s}} \int_0^s \frac{\mathrm{d}s}{\dot{s}} = \min_{\dot{u}} \int_0^1 \frac{\mathrm{d}u}{\dot{u}} \tag{14}$$

where s is the length of the parameterized curve.

For problem (14), minimizing flight time is equivalent to maximize the velocity at each point on the curve, which is [43]

$$\begin{array}{ll} \max\ \mathcal{J} = \sum_{i=1}^{T_N} \xi_i & \boldsymbol{\xi} = [\xi_0, \xi_1, \xi_2, \cdots, \xi_n]^T,\ \xi = V^2(u_i) \\ s.t.\ \ \mathbf{A}\boldsymbol{\xi} \le \mathbf{b},\ \xi_i \ge 0 & \end{array} \tag{15}$$

where $u_i = i/T_N$.

Denoting $\kappa = \sqrt{x_u(u)^2 + y_u(u)^2 + z_u(u)^2}$, where $(\cdot)_u$ is the derivative of the corresponding variable to u, one can obtain

$$\left\| \frac{\mathrm{d}\mathbf{C}(u)}{\mathrm{d}u} \right\| = \left\| \frac{\mathrm{d}\mathbf{C}(u)}{\mathrm{d}s} \frac{\mathrm{d}s}{\mathrm{d}u} \right\| = \kappa \tag{16}$$

Therefore, (9) can be rewritten as

$$\frac{\mathrm{d}u}{\mathrm{d}t} = \frac{V(u)}{\kappa} \tag{17}$$

In such a case, $\ddot{u}$ can be expressed as

$$\frac{\mathrm{d}^2u}{\mathrm{d}t^2} = \frac{\mathrm{d}(V(u)/\kappa)}{\mathrm{d}t} = a(u) \tag{18}$$

where

$$a(u) = \frac{\mathrm{d}V(u)}{\mathrm{d}t} = (V(u)/\kappa)_u V(u)/\kappa = \frac{1}{2}\frac{\mathrm{d}(V(u)/\kappa)^2}{\mathrm{d}u} \tag{19}$$

which can be rewritten in a discrete form as

$$\begin{aligned} a(u_i) = {} & \mathbf{C}^{\mu}_{uu}(u_i)\left(\tfrac{V(u_i)}{\kappa(u_i)}\right)^2 \\ & + \frac{\mathbf{C}^{\mu}_{u}(u_i)\left((\frac{V(u_{i+1})}{\kappa(u_{i+1})})^2 - (\frac{V(u_{i-1})}{\kappa(u_{i-1})})^2 \right)}{2(u_{i+1} - u_{i-1})} \end{aligned} \tag{20}$$

In the real-time trajectory tracking process, the tracking error will increase with the flight speed. This error is also named as chord error, which can be estimated as follows

$$\varepsilon_i = \rho_i - \sqrt{\rho_i^2 - \left(\frac{V(u_i)T_s}{2}\right)} \tag{21}$$

For a given maximum chord error $\varepsilon_{\max}$, the flight speed constraint should be introduced in the following way [42,43]

$$V(u_i) \le 2\sqrt{\rho_i^2 - (\rho_i - \varepsilon_{\max})^2}/T_s \tag{22}$$

where ρ_i can be calculated with the following formula

$$\rho(u|_{u=u_i}) = \frac{\|C_u(u)\|^3}{\|C_u(u) \times C_{uu}(u)\|} \tag{23}$$

When expressed in the three dimensional coordinates x, y, and z, (23) can be explicitly rewritten as follows

$$\rho|_{u=u_i} = \frac{[(x_u)^2 + (y_u)^2 + (z_u)]^{\frac{3}{2}}}{\left[\begin{vmatrix} y_u & z_u \\ y_{uu} & z_{uu} \end{vmatrix} + \begin{vmatrix} z_u & x_u \\ z_{uu} & x_{uu} \end{vmatrix} + \begin{vmatrix} x_u & y_u \\ x_{uu} & y_{uu} \end{vmatrix}\right]^{1/2}} \tag{24}$$

Neglecting the second order small term, (22) can be reduced as

$$V(u_i) \le 2\sqrt{2\rho_i\varepsilon_{\max}}/T_s \tag{25}$$

On the other hand, for each component of $\mathbf{C}(u)$, denoted as $\mathbf{C}^\mu$ ($\mu \in \{x,y,z\}$), its velocity and acceleration can be estimated as

$$\begin{cases} V^\mu(u) = \frac{\mathrm{d}\mathbf{C}^\mu(u)}{\mathrm{d}t} = \mathbf{C}_u^\mu \frac{\mathrm{d}u}{\mathrm{d}t} = \mathbf{C}_u^\mu \frac{V(u)}{\kappa} \\ A^\mu(u) = \frac{\mathrm{d}V^\mu(u)}{\mathrm{d}u} = \mathbf{C}_{uu}^\mu \left(\frac{V(u)}{\kappa}\right)^2 + \mathbf{C}_u^\mu a(u) \end{cases} \tag{26}$$

In this way, the discrete form of the acceleration constraint is

$$\left|\mathbf{C}_{uu}^\mu(u_i)\left(\frac{V(u_i)}{\kappa}\right)^2 + \frac{\mathbf{C}_u^\mu(u_i)\left(\frac{V(u_{i+1})^2}{\kappa(u_{i+1})^2} - \frac{V(u_{i-1})^2}{\kappa(u_{i-1})^2}\right)}{2(u_{i+1}-u_{i-1})}\right| \le A_{\max}^\mu \tag{27}$$

Alternatively, the acceleration of each dimension can be expressed in the following way

$$\begin{aligned} A^\mu(u) &= \frac{\mathbf{d}V^\mu(u)}{\mathbf{d}t} = \frac{\mathbf{d}V^\mu(u)}{\mathbf{d}\mathbf{C}^\mu(u)}\frac{\mathrm{d}\mathbf{C}^\mu}{\mathrm{d}t} \\ &= V^\mu(u)\frac{\mathbf{d}V^\mu(u)}{\mathbf{d}\mathbf{C}^\mu(u)} = \frac{1}{2}\frac{\mathrm{d}(\mathbf{C}_u^\mu(u)^2 V^\mu(u)^2/\kappa^2)}{\mathrm{d}\mathbf{C}^\mu(u)} \end{aligned} \tag{28}$$

which can be written in a discrete form as

$$\begin{aligned} &\left|\frac{\mathbf{C}_u^\mu(u_{i+1})^2}{\kappa(u_{i+1})^2}V^\mu(u_{i+1})^2 - \frac{\mathbf{C}_u^\mu(u_i)^2}{\kappa(u_i)^2}V^\mu(u_i)^2\right| \\ &\qquad \le 2A_{\max}^\mu|\mathbf{C}^\mu(u_{i+1}) - \mathbf{C}^\mu(u_i)| \end{aligned} \tag{29}$$

The results calculated from (29) and (27) are the same. However, as the discrete interval of (29) is smaller than that of (27), (29) could be more accurate.

Supposing the velocities at the start point and the end point are 0, the constraints in (15) can be expressed as

$$\begin{aligned} &\max\ \mathcal{J} = \textstyle\sum_{i=1}^{T_N} V^2(u_i) \\ &s.t.\quad 0 \le V^2(u_i) \le \min\{V_{\max}^2, 8\rho_i\varepsilon_{\max}/T_s^2\} \\ &\qquad V^2(u_0) = V^2(u_{T_N}) = 0 \\ &\qquad |\mathbf{C}_{uu}^\mu(u_i)\left(\frac{V(u_i)}{\kappa(u_i)}\right)^2 + \frac{\mathbf{C}_u^\mu(u_i)\left((\frac{V(u_{i+1})}{\kappa(u_{i+1})})^2 - (\frac{V(u_{i-1})}{\kappa(u_{i-1})})^2\right)}{2(u_{i+1}-u_{i-1})}| \\ &\qquad\qquad \le A_{\max}^\mu \quad (\mu \in \{x,y,z\}) \end{aligned} \tag{30}$$

The optimal velocity sequence can be obtained by solving (30), and the trajectory reference is then calculated by substituting the velocity sequence into (13).

However, there could exist rapid changes in the acceleration considering the solution of (30), which leads to instability of the system. To tackle this problem, the jerk constraints are introduced as follows. For the sake of simplicity, an intermediate variable $q(u) = V(u)^2/\kappa^2$ is defined in a similar way as in [42]. Denoting $q(u)$ as q and denoting $q(u_i)$ as q_i, (26) can be rewritten as [42]

$$A^\mu(u) = \mathbf{C}_{uu}^\mu q + \mathbf{C}_u^\mu q_u/2 \tag{31}$$

In this way, the jerk of the NURBS curve can be expressed as [42]

$$J^\mu(u) = \dot{A}^\mu(u) = \left(\mathbf{C}_{uuu}^\mu q + \frac{3\mathbf{C}_{uu}^\mu}{2}q_u + \frac{\mathbf{C}_u^\mu}{2}q_{uu}\right)\sqrt{q} \tag{32}$$

Denoting $\Delta u = u_{i+1} - u_i$ and considering $q_u \approx (q_{i+1} - q_{i-1})/(2\Delta u)$ and $q_{uu} \approx \frac{q_{i+1}+q_{i-1}-2q_i}{\Delta u^2}$, (32) can be rewritten in the following form

$$\left|(\alpha_i^\mu q_{i-1} + \beta_i^\mu q_i + \gamma_i^\mu q_{i+1})\right|\sqrt{q_i} \le J_{\max} \tag{33}$$

where $\alpha_i^\mu = \frac{\mathbf{C}_u^\mu(u_i)}{2\Delta u^2} - \frac{3\mathbf{C}_{uu}^\mu(u_i)}{4\Delta u}$, $\beta_i^\mu = \mathbf{C}_{uuu}^\mu(u_i) - \frac{\mathbf{C}_u^\mu(u_i)}{\Delta u^2}$, $\gamma_i^\mu = \frac{\mathbf{C}_u^\mu(u_i)}{2\Delta u^2} + \frac{3\mathbf{C}_{uu}^\mu(u_i)}{4\Delta u}$.

Specifically, the jerk at the start point and the end point can be estimated as

$$J^\mu = \frac{\mathrm{d}A^\mu}{\mathrm{d}t} = A^\mu\frac{\mathrm{d}A^\mu}{\mathrm{d}V^\mu} = \frac{1}{2}\frac{\mathrm{d}(A^\mu)^2}{\mathrm{d}V} \tag{34}$$

In this way, for the first interval $[0,u_1]$, $|J_1^\mu| = |\left((A_1^\mu)^2 - (A_0^\mu)^2\right)/(2V_1^\mu - 2V_0^\mu| = |(A_1^\mu)^2/(2\mathbf{C}_u^\mu\sqrt{q_1})| =$

$\left|T_N^2(\mathbf{C}_u^\mu(u_2))^2\sqrt{q_1}q_2/(8\mathbf{C}_u^\mu(u_1))\right|$. Similarly, for the last interval $|J_{T_N}^\mu| = \left|T_N^2(\mathbf{C}_u^\mu(u_{N-2}))^2\sqrt{q_{N-1}}q_{N-2}/(8\mathbf{C}_u^\mu(u_{N-1}))\right|$. In this way, the jerk constraints in the first and last intervals are rewritten as follows

$$\begin{aligned}&\left|T_N^2(\mathbf{C}_u^\mu(u_2))^2\sqrt{q_1}q_2/(8\mathbf{C}_u^\mu(u_1))\right| \le J_{\max}\\&\left|T_N^2(\mathbf{C}_u^\mu(u_{N-2}))^2\sqrt{q_{N-1}}q_{N-2}/(8\mathbf{C}_u^\mu(u_{N-1}))\right| \le J_{\max}\end{aligned} \tag{35}$$

Obviously, the constraints in (33) and (35) are nonlinear. To enhance the computational efficiency, they are linearized as follows. Supposing the solution of (30) is $\{q_i^*\}$, one can multiply $\sqrt{\frac{q_i^*}{q_i}}$ on both side of (35) and the following relationship can be established

$$\begin{aligned}&\left|(\sqrt{q_i^*}\alpha_i^\mu q_{i-1} + \sqrt{q_i^*}\beta_i^\mu q_i + \sqrt{q_i^*}\gamma_i^\mu q_{i+1})\right|\\&\qquad \le J_{\max}\sqrt{\frac{q_i^*}{q_i}}\end{aligned} \tag{36}$$

$$\begin{aligned}&\left|T_N^2(\mathbf{C}_u^\mu(u_2))^2\sqrt{q_1^*}q_2/(8\mathbf{C}_u^\mu(u_1))\right| \le J_{\max}\sqrt{\frac{q_i^*}{q_i}}\\&\left|T_N^2(\mathbf{C}_u^\mu u_{N-2}))^2(\sqrt{q_{N-1}^*}q_{N-2}/(8\mathbf{C}_u^\mu(u_{N-1}))\right|\\&\qquad \le J_{\max}\sqrt{\frac{q_i^*}{q_i}}\end{aligned} \tag{37}$$

Considering the inequality $\sqrt{\frac{q_i^*}{q_i}} \ge \frac{3}{2} - \frac{q_i}{2q_i} \ge 1$ [42], if q_i satisfy following constrains, then they also satisfy the constraints in (37)

$$\begin{aligned}&\left|(\sqrt{q_i^*}\alpha_i^\mu q_{i-1} + \sqrt{q_i^*}\beta_i^\mu q_i + \sqrt{q_i^*}\gamma_i^\mu q_{i+1})\right|\\&\qquad \le J_{\max}\left(\frac{3}{2} - \frac{q_i}{2q_i^*}\right)\end{aligned} \tag{38}$$

$$\begin{aligned}&\left|T_N^2(\mathbf{C}_u^\mu(u_2))^2\sqrt{q_1^*}q_2/(8\mathbf{C}_u^\mu(u_1))\right| \le J_{\max}\left(\frac{3}{2} - \frac{q_1}{2q_1^*}\right)\\&\left|T_N^2(\mathbf{C}_u^\mu(u_{N-2}))^2\sqrt{q_{N-1}^*}q_{N-2}/(8\mathbf{C}_u^\mu(u_{N-1}))\right|\\&\qquad \le J_{\max}\left(\frac{3}{2} - \frac{q_{N-1}}{2q_{N-1}^*}\right)\end{aligned} \tag{39}$$

Therefore, the optimal velocity sequence can be solved in two steps. Firstly, the optimal solution of (30) can be obtained without the constraints on the jerk. Then the jerk constraints are introduced into (39) to obtain the final results.

3.4 Complexity Analysis

The computational complexity of the trajectory generation approach proposed by this work concerns three components: the geometric trajectory generation with the least square method or damped least square method, the linear programing with acceleration constraints, and the linear programing with the jerk constraints. As the number of the desired navigation points is commonly much smaller than that of the discrete points in the velocity planning, its computational cost can be neglected. The linear programing problem in the velocity planning is of fixed T_N dimension, and the number of constraints involved is also linearly related to T_N. Therefore, using the method proposed by Megiddo [46], it can be solved in $O(n)$ time. In practice, some other method like the active set algorithm can also achieve $O(n)$ computational complexity [47]. This implies with n sampling points in the two-step optimization method, $O(n)$ computational efficiency is guaranteed.

4 Trajectory tracking controller

The trajectory tracking controller is designed based on the subspace stabilization approach [10].

Firstly, regarding the desired reference trajectory $\mathbf{C}^*(t)$, a base tracking controller can be designed based on the differential flatness of the quadrotor $\mathbf{v}_b \triangleq \{G_1, \theta, \phi\} = \lambda_p(\mathbf{C}^*(t))$. Then denoting the position tracking error and the velocity tracking error by $\mathbf{e}_p$ and $\mathbf{e}_v$, a compensation input item is designed as

$$\mathbf{v}_c \triangleq \{\delta U_1, \delta\theta, \delta\phi\} = \mathbf{k}_p \circ \mathbf{e}_p + \mathbf{k}_v \circ \mathbf{e}_v \tag{40}$$

where the operator $\circ$ stands for the Hadamard product [48], i.e., $\mathbf{c} = \mathbf{a} \circ \mathbf{b}$ is defined such that the ith element $\mathbf{c}_i$ equals to the production of the ith elements $\mathbf{a}_i$ and $\mathbf{b}_i$.

In this way, the trajectory tracking controller is designed as

$$\mathbf{v} = \mathbf{v}_b + \mathbf{v}_c \tag{41}$$

The control input is then sent to the attitude control loop, which is designed as an inner loop based on the proportional-derivative (P-D) control technique [19].

5 Simulations and Experiments

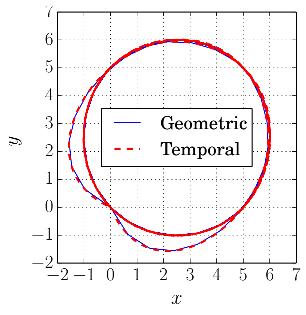
In this section, comparative numeric simulations regarding representative existing methods are firstly carried out to verify the efficiency and effectiveness of the proposed approach. Then, real time experiments are conducted to further verify the proposed trajectory generation approach and tracking controller.

5.1 Algorithm Implementation

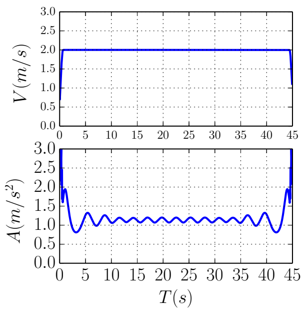
The trajectory optimization approach proposed by this work aims to generate optimal reference sequence based on the desired way-points, therefore the inputs for the algorithm are the desired way-points, and the output of this algorithm is the time-optimal trajectory reference. Based on the development in Section 3, the algorithm itself can be implemented in five steps as follows:

Step 1): Evaluate the geometric trajectory with known desired way-points by solving the optimum problem (6).

Step 2): Discretize the parameter u into an T_Nth order vector.



(a) Optimal trajectory



(b) Velocity and acceleration evolutions

Fig. 2. The trajectory optimization for the trajectory with multiple squares.

Step 3): Solve the linear programming problem (30) imposed with the acceleration constraints.

Step 4): Solve the linear programming problem (39) imposed with the jerk constraints and obtain the optimal velocity reference sequence.

Step 5): Substitute the obtained velocity sequence into (13) and establish the desired optimal reference trajectory.

5.2 Numeric Verification

As the real-time trajectory generation for autonomous navigation in the unknown cluttered environment is well verified in [31] recently, a comparative simulation is thereby carried out in this work based on the example given by [31], which aims to obtain the optimal trajectory passing through vertexes aligned in multiple (four) 5×5 squares. The proposed algorithms are developed with MATLAB R2014 on a laptop [Intel Core (TM) i5-5200 CPU @2.20GHZ$\times$4, 8GB Memory, OS: Ubuntu 14.04].

As shown in Fig. 2(a), the optimal trajectory generated by this work is similar with that in [31], which roughly establishes multiple circles passing the vertexes. When $T_N = 200$, the computational time for the trajectory optimization is about 2.5 seconds, and when $T_N = 100$, which generate almost the same trajectory profile, the total computational time is about 0.8 second. This computational time is even slightly less than that in [31] for the optimization of a trajectory with 10 segments (In this work, there are 16 segments in the trajectory of multiple squares). As the computation platform in this work might be much different with that in [31], and also the computational time both in [31] and in this work are well suited for real-time trajectory generation, the significance in the timing comparison might be not so practically important.

However, when examining Fig. 2(b), it can be clearly found that the trajectory generated by this work is much more efficient. When the velocity constraint $V_{max} = 2$ m/s and acceleration constraint $A_{max} = 2\ \mathrm{m/s^2}$ are imposed, the optimal trajectory generated by [31] requires approximately 112 s to fulfill entire flight. In this work, with additional jerk constraint $J^{\mu}_{\max} = 10\ \mathrm{m/s^3}$ and chord error constraint $\varepsilon_{\max} = 0.05$ m, the entire flight only takes about 45 s. In Fig. 2(b), the evolution shows steady velocity $V = 2$ m/s in the whole process except at the acceleration and deceleration stage. Similarly, the acceleration mostly evolutes near $A = 1\ \mathrm{m/s^2}$, except at the start and end point, where the acceleration exceeds the constraints due to the numeric inaccuracy. Compared to the results in [31] where the acceleration and velocity may constantly exceeds the constraint for about 10%, the results shown in Fig. 2 are also effective.

In summery, although the optimization for jerk is not considered in this work, the constraint for jerk is imposed alternatively. In this way, the optimality might be sacrificed to an extent, but the overall efficiency and effectiveness is actually improved compared to the representative real-time trajectory generation approaches in the autonomous navigation. In addition, it is worth noting that the nonlinear optimal problems in [31] cannot be 100% solved. In this sense, the linear programming method proposed in this work also presents better reliability.

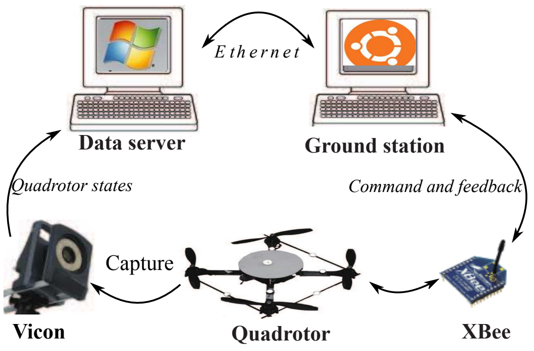


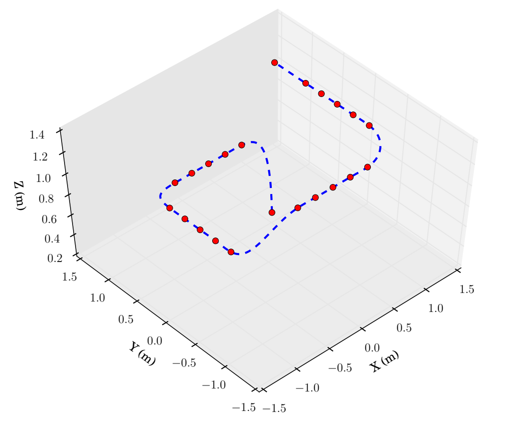
Fig. 3. The quadrotor test bed.

5.3 Experimental Setup

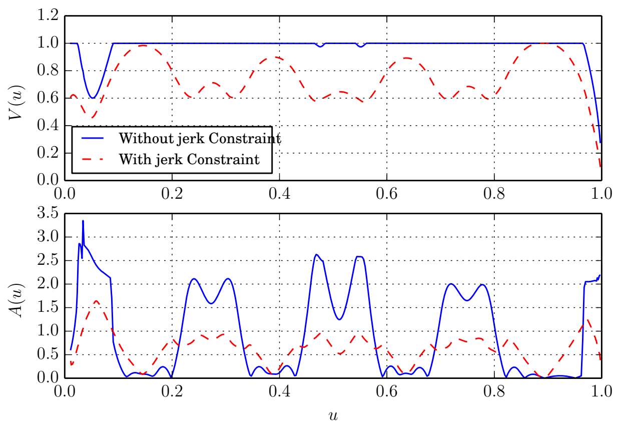
The experiments are conducted in an indoor test bed, as shown in Fig. 3, where a commercial quadrotor platform, namely the Hummingbird quadrotor, is adopted. The quadrotor communicates with a ground control station via the XBee wireless routers at a frequency of 50 Hz. The attitude of the quadrotor is stabilized by the on-board P-D controller written in C-codes, and both of the trajectory generation and position tracking controller of the quadrotor are implemented on the ground control station. To improve the real-time performance of the flight control, the ground control station runs on a Linux (Ubuntu 14.04) operating system and the trajectory generation and tracking algorithms are developed in the form of a Simulink model. An S-function in control station polls the position state of the quadrotor from the data sever of the Vicon motion capture system, and the velocity of the quadrotor is then obtained by taking derivative of the position data. The motion capture system runs at a frequency of 200 Hz, and the motion of quadrotor is captured by kinematic fitting of the reflective markers attached on the quadrotor. More details of this test bed can be found in [27].

5.4 Experimental Results

Considering the cluttered environment presented in [27], the following 3D points are adopted to simulate the desired

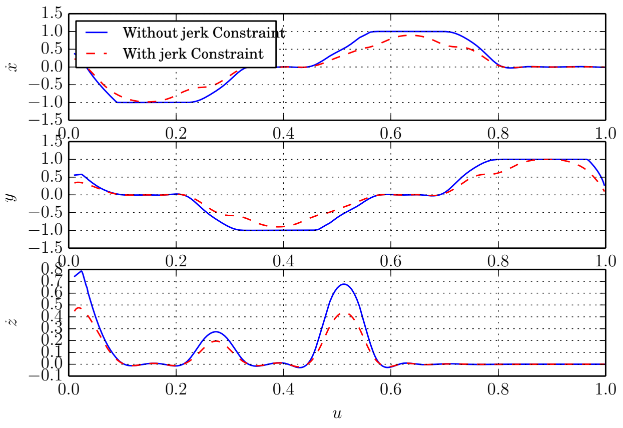


(a) 4th order NUBRS trajectory for ring obstacle avoidance

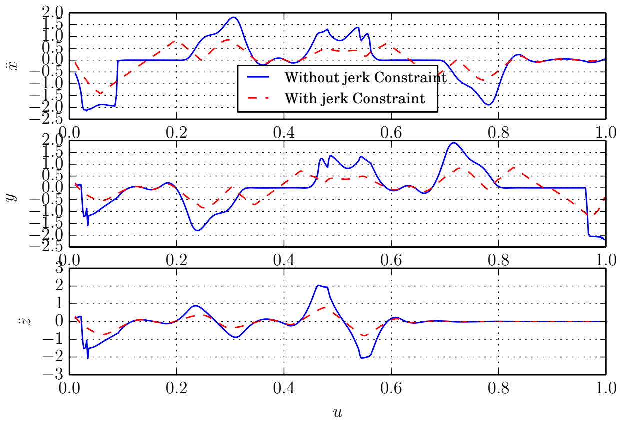


(b) velocities and accelerations

Fig. 4. The 4th NURBS trajectory regarding a cluttered environment with three rings.

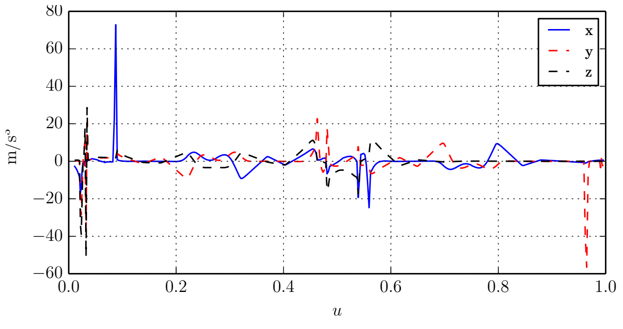


(a) Velocity evolution

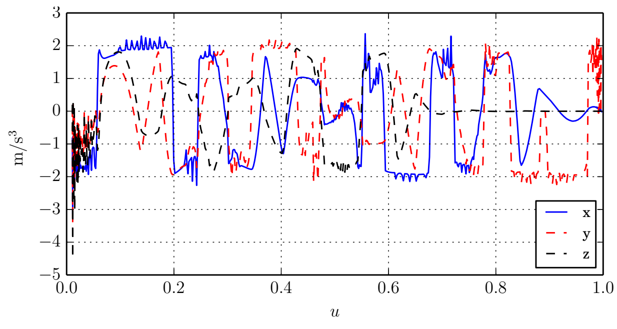


(b) Acceleration evolution

Fig. 5. The velocity and acceleration evolution of the 4th o NURBS trajectory.



(a) Without jerk constraints

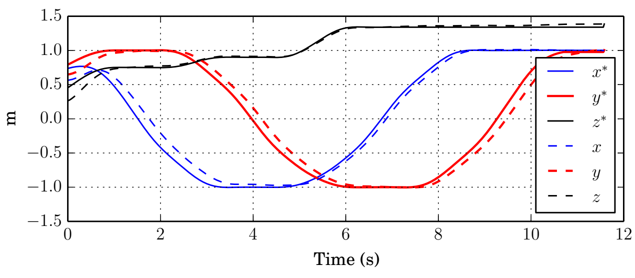


(b) With jerk constraints

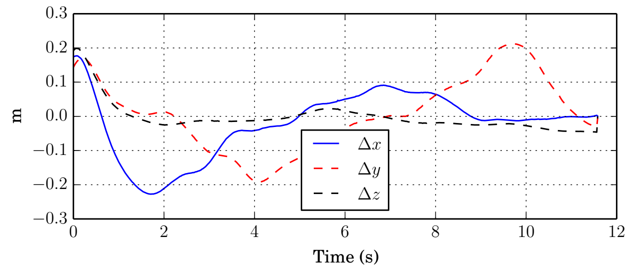
Fig. 6. The jerk evolution of the 4th order NURBS trajectory.

navigation way-points in an autonomous flight under unknown environment: [0.6, 0.6, 0.2; 0.5, 1, 0.75; 0.25, 1, 0.75; 0, 1, 0.75; -0.25, 1, 0.75; -0.5, 1, 0.75; -1, 0.5, 0.9; -1, 0.25, 0.9; -1, 0, 0.9; -1, -0.25, 0.9; -1, -0.5, 0.9; -0.5, -1, 1.34; -0.25, -1, 1.34; -0, -1, 1.34; 0.25, -1, 1.34; 0.5, -1, 1.34; 1, -0.5, 1.34; 1, -0.25, 1.34; 1, 0, 1.34; 1, 0.25, 1.34; 1, 0.5, 1.34; 1, 1, 1.34;].

Based on the given way-points, a 4th order geometric trajectory based on the NURBS curve can be generated as shown in Fig. 4. Then with sampling number $T_N = 500$, chord error constraint $\varepsilon_{\max} = 0.02$ m, velocity constraint $V_{\max} = 1$ m/s, acceleration constraint $A^{\mu}_{\max} = 2$ m/s^2, and jerk constraint $J^{\mu}_{\max} = 2$ m/s^3, the time-optimal trajectory can be generated based on the algorithm described in Section 5.1. The corresponding velocity and acceleration evolutions are shown in Fig. 5. It can be seen that the accelerations in the x, y, and z directions are less than 2 m/s^2. This indicates that the accelerations of the flight are moderate. In this way, the quadrotor can provide sufficient thrust for the corresponding movement. When the jerk constraints are introduced, the profiles of the velocity and acceleration are more smooth, thus guarantees more stable flights. The components of the velocity and acceleration in each direction

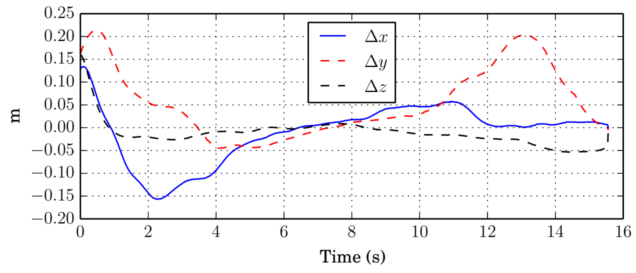


(a) Trajectory tracking process

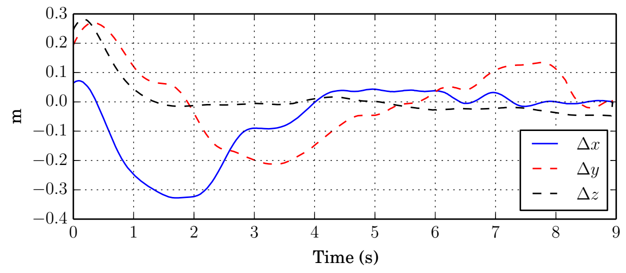


(b) Dynamic error

Fig. 7. The trajectory tracking process when $V_{\max} = 1\text{m/s}$, $A_{\max} = 2\text{m/s}^2$, $J_{\max} = 2\text{m/s}^3$



(a) $J_{\max} = 1\text{m/s}^3$



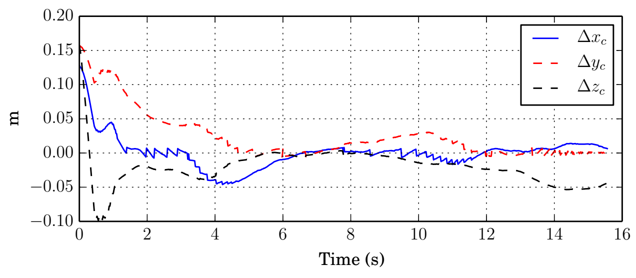
(b) $J_{\max} = 5\text{m/s}^3$

Fig. 8. The along-tracking error when with different jerk constraint.

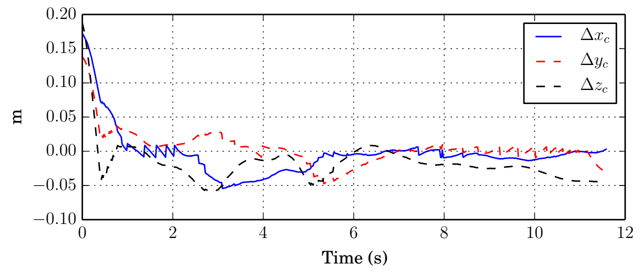
of this process are also shown in Fig. 5, where all of the variables smoothly evolutes within the pre-defined constraints. In addition, the jerk evolution is demonstrated in Fig. 6. It can be seen that when without constraints, there exist rapid changes in the jerk evolution, as shown in Fig. 6(a). In contrast, when the constraint $J^{\mu}_{\max} = 2\ \text{m/s}^3$ is imposed in Fig. 6(b), the jerk is constrained in the desired range. It should be noted that due to the numeric inaccuracy, there exist small fluctuations in the jerk evolution.

Utilizing the trajectory tracking controller developed in Section 4 for trajectory tracking, and adopting $\mathbf{k}_p = [1.5, 1.5, 5]^T$ and $\mathbf{k}_v = [1.2, 1.2, 2.5]^T$, which are tuned by the trial and error method, the trajectory tracking process is indicated in Fig. 7. It is shown that the quadrotor can closely track the reference trajectory, except at the very beginning. This is because the quadrotor doesn't start the trajectory tracking process at the exact location of the starting point of the desired trajectory, and also there are ground effects considering the aerodynamic effects. Nonetheless, in this way the performance and robustness of the trajectory tracking controller can be demonstrated more convincingly. This can be seen from Fig. 7 that the along-tracking errors roughly decrease with time, and this phenomenon is more evident in the cross-tracking errors which will be discussed in the following text. The along-tracking errors regarding reference trajectories with different jerk constraints also demonstrate similar behaviors, as shown in Fig. 8.

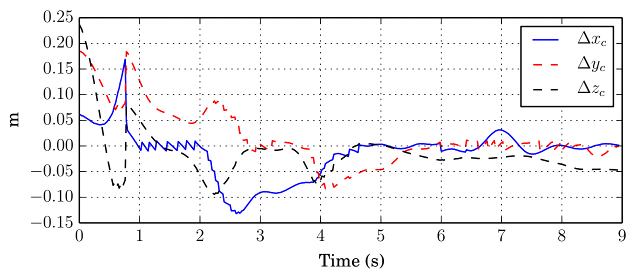
The cross-tracking errors in all of these processes are reasonably small, which are shown in Fig. 9. As mentioned, due to the ground effects and initial tracking error, the cross-tracking errors at the beginning seem significant. However, with the tracking controller, the cross-tracking errors decrease with time in all of the three cases $J^{\mu}_{\max} = 1\ \text{m/s}^3$, $J^{\mu}_{\max} = 2\ \text{m/s}^3$, and $J^{\mu}_{\max} = 5\ \text{m/s}^3$. Furthermore, the cross-tracking errors in x, y, and z directions are less than 0.05 m when $t > 5$ s.



(a) $J_{\max} = 1\text{m/s}^3$



(b) $J_{\max} = 2\text{m/s}^3$



(c) $J_{\max} = 5\text{m/s}^3$

Fig. 9. The cross-tracking error when with different jerk constraint.

All of these results indicate that the proposed trajectory generation approach can guarantee time-optimal reference trajectory for the autonomous navigation, and the developed tracking controller can closely track the generated trajectory. In this way, when the navigation way-points are known, the desired smooth reference trajectory can be obtained and effectively tracked.

6 Conclusion

A time-optimal trajectory generation approach, which is suitable for the real-time path planning for the fully autonomous navigation in cluttered environment, is proposed for the quadrotor based on the NURBS curve and linear programming. This approach first constructs the desired geometric trajectory based on multiple way-point navigation and the NURBS curve. With the geometric trajectory, a time-optimal interpolation problem is then imposed considering the velocity, acceleration and jerk constraints. The optimization problem is solved in two steps. At the first step, a preliminary result is obtained by solving a linear programming without jerk constraints. Subsequently, substituting the obtained results and properly relaxing the jerk constraints, an additional linear programming problem is introduced and the original time-optimal problem can be fully solved. This trajectory generation is straightforward and computationally efficient considering the existing optimization theory. To well track the generated reference trajectories, a nonlinear trajectory tracking controller considering the differential flatness property of the quadrotor is developed as well.

The feasibilities of the proposed trajectory generation approach as well as the tracking controller are verified through both numeric simulations and real-time experiments. In the simulations, the proposed approach demonstrates both computational efficiency and trajectory time-optimality compared to the representative existing methods. In addition, although the minimum jerk or snap objectives are not imposed in this work, compared to the existing works, the generated trajectory still shows satisfactory dynamic performance since velocity, acceleration and jerk constraints are properly introduced instead. In real-time experiments, the quadrotor is tasked to fly through a cluttered indoor environment with ring obstacles. The experimental results show that with velocity, acceleration, and jerk constraints: $V_{\max} = 1$ m/s, $A_{\max} = 2$ m/s^2, and $J_{\max} = 5$ m/s^3, the proposed approach can generate trajectory with smooth acceleration profile and moderate velocity $v \approx 1$ m/s, which is consistent with dynamics of the quadrotor. In such a case, the trajectory tracking controller can closely track the reference trajectory with the cross-tracking errors less than 0.05 m. All of the results demonstrate that the proposed trajectory generation approach, owing to its straightforwardness and computational efficiency, can well construct dynamically consistent trajectories for the quadrotor to navigate through multiple way-points in real-time applications.

References

[1] Ijspeert, A. J., 2014. "Biorobotics: Using robots to emulate and investigate agile locomotion". *Science,* **346**(6206), pp. 196–203.

[2] Lindsey, Q., Mellinger, D., and Kumar, V., 2012. "Construction with quadrotor teams". *Autonomous Robots,* **33**(3), pp. 323–336.

[3] Huang, P.-Y., Zeng, L.-R., Yang, C., Peng, Y.-X., and Yu, C.-B., 2014. "Design of a new style four-axis aerial photography aircraft for disaster relief". *Journal of Sichuan Ordnance,* **6**, p. 035.

[4] Abdolhosseini, M., Zhang, Y., and Rabbath, C. A., 2013. "An efficient model predictive control scheme for an unmanned quadrotor helicopter". *Journal of Intelligent & Robotic Systems,* **70**(1-4), pp. 27–38.

[5] Lim, H., Park, J., Lee, D., and Kim, H., 2012. "Build your own quadrotor: Open-source projects on unmanned aerial vehicles". *IEEE Robotics & Automation Magazine,* **19**(3), pp. 33–45.

[6] Doyle, C. E., Bird, J. J., Isom, T. A., Kallman, J. C., Bareiss, D. F., Dunlop, D. J., King, R. J., Abbott, J. J., Minor, M., et al., 2013. "An avian-inspired passive mechanism for quadrotor perching". *IEEE/ASME Transactions on Mechatronics,* **18**(2), pp. 506–517.

[7] Ding, X., and Yu, Y., 2013. "Motion planning and stabilization control of a multipropeller multifunction aerial robot". *IEEE/ASME Transactions on Mechatronics,* **18**(2), pp. 645–656.

[8] Muller, M., Lupashin, S., and D'Andrea, R., 2011. "Quadrocopter ball juggling". In Proceedings of IEEE International Conference on Intelligent Robots and Systems, pp. 5113–5120.

[9] Dong, W., Gu, G.-Y., Zhu, X., and Ding, H., 2015. "Solving the boundary value problem of an under-actuated quadrotor with subspace stabilization approach". *Journal of Intelligent & Robotic Systems,* **80**(2), November, pp. 299–311.

[10] Dong, W., Gu, G.-Y., Ding, Y., Xiangyang, Z., and Ding, H., 2015. "Ball juggling with an under-actuated flying robot". In Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 68–73.

[11] Hehn, M., and D'Andrea, R., 2011. "A flying inverted pendulum". In Proceedings of IEEE International Conference on Robotics and Automation, pp. 763–770.

[12] Bezzo, N., Griffin, B., Cruz, P., Donahue, J., Fierro, R., and Wood, J., 2014. "A cooperative heterogeneous mobile wireless mechatronic system". *IEEE/ASME Transactions on Mechatronics,* **19**(1), pp. 20–31.

[13] Mellinger, D., and Kumar, V., 2011. "Minimum snap trajectory generation and control for quadrotors". In Proceedings of IEEE International Conference on Robotics and Automation, pp. 2520–2525.

[14] Kumar, V., and Michael, N., 2012. "Opportunities and challenges with autonomous micro aerial vehicles". *The International Journal of Robotics Research,* **31**(11), pp. 1279–1291.

[15] Huang, H., Hoffmann, G. M., Waslander, S. L., and

Tomlin, C. J., 2009. "Aerodynamics and control of autonomous quadrotor helicopters in aggressive maneuvering". In Proceedings of the IEEE International Conference on Robotics and Automation, pp. 3277–3282.

[16] Cutler, M., and How, J. P., 2015. "Analysis and control of a variable-pitch quadrotor for agile flight". *Journal of Dynamic Systems, Measurement, and Control,* **137**(10), p. 101002.

[17] Do, K. D., 2015. "Path-tracking control of stochastic quadrotor aircraft in three-dimensional space". *Journal of Dynamic Systems, Measurement, and Control,* **137**(10), p. 101003.

[18] Goodarzi, F. A., Lee, D., and Lee, T., 2015. "Geometric adaptive tracking control of a quadrotor unmanned aerial vehicle on se (3) for agile maneuvers". *Journal of Dynamic Systems, Measurement, and Control,* **137**(9), p. 091007.

[19] Dong, W., Gu, G.-Y., Zhu, X., and Ding, H., 2014. "High-performance trajectory tracking control of a quadrotor with disturbance observer". *Sensors and Actuators A: Physical,* **211**, pp. 67 – 77.

[20] Cichella, V., Kaminer, I., Dobrokhodov, V., Xargay, E., Choe, R., Hovakimyan, N., Pascoal, A. M., et al., 2015. "Cooperative path following of multiple multirotors over time-varying networks". *IEEE Transactions on Automation Science and Engineering,* **12**(3), pp. 945–957.

[21] Mellinger, D., Michael, N., and Kumar, V., 2012. "Trajectory generation and control for precise aggressive maneuvers with quadrotors". *The International Journal of Robotics Research,* **31**(5), pp. 664–674.

[22] Mueller, M. W., Hehn, M., and D'Andrea, R., 2013. "A computationally efficient algorithm for state-to-state quadrocopter trajectory generation and feasibility verification". In Proceedings of IEEE International Conference on Intelligent Robots and Systems, pp. 3480–3486.

[23] Bangura, M., and Mahony, R., 2012. "Nonlinear dynamic modeling for high performance control of a quadrotor". In Proceedings of the Australasian conference on robotics and automation, pp. 1–10.

[24] Hehn, M., and DAndrea, R., 2011. "Quadrocopter trajectory generation and control". In Proceedings of the IFAC World Congress, Vol. 18, pp. 1485–1491.

[25] Bouktir, Y., Haddad, M., and Chettibi, T., 2008. "Trajectory planning for a quadrotor helicopter". In Proceedings of the Mediterranean Conference onthe Control and Automation, pp. 1258–1263.

[26] Cowling, I., 2008. "Towards autonomy of a quadrotor uav". PhD thesis, Cranfield University, Bedfordshire.

[27] Dong, W., Gu, G., Zhu, X., and Ding, H., 2015. "Development of a quadrotor test bed–modelling, parameter identification, controller design and trajectory generation". *Int J Adv Robot Syst,* **12**, p. 7.

[28] Nuske, S., Choudhury, S., Jain, S., Chambers, A., Yoder, L., Scherer, S., Chamberlain, L., Cover, H., and Singh, S., 2015. "Autonomous exploration and motion planning for an unmanned aerial vehicle navigating rivers". *Journal of Field Robotics,* **32**(8), pp. 1141–1162.

[29] Droeschel, D., Nieuwenhuisen, M., Beul, M., Holz, D., Stückler, J., and Behnke, S., 2015. "Multilayered mapping and navigation for autonomous micro aerial vehicles". *Journal of Field Robotics*. Article first published online: 5 JUN 2015, DOI: 10.1002/rob.21603.

[30] Karaman, S., and Frazzoli, E., 2011. "Sampling-based algorithms for optimal motion planning". *The International Journal of Robotics Research,* **30**(7), pp. 846–894.

[31] Burri, M., Oleynikova, H., Achtelik, M., and Siegwart, R., 2015. "Real-time visual-inertial mapping, re-localization and planning onboard mavs in unknown environments". In Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 1872–1878.

[32] Richter, C., Bry, A., and Roy, N., 2013. "Polynomial trajectory planning for aggressive quadrotor flight in dense indoor environments". In Proceedings of the International Symposium on Robotics Research, pp. 1–16.

[33] Hoffmann, G. M., Waslander, S. L., and Tomlin, C. J., 2008. "Quadrotor helicopter trajectory tracking control". In Proceedings of the AIAA Guidance, Navigation and Control Conference and Exhibit, pp. 1–14.

[34] Mathew, N., Smith, S. L., and Waslander, S. L., 2014. "Optimal path planning in cooperative heterogeneous multi-robot delivery systems". In Workshop on the Algorithmic Foundations of Robotics, Istanbul, Turkey.

[35] Kuo, C., Kuo, C., Leber, A., and Boller, C., 2013. "Vector thrust multi-rotor copter and its application for building inspection". *International Micro Air Vehicle Conference and Flight Competition*.

[36] Benson, D. A., Huntington, G. T., Thorvaldsen, T. P., and Rao, A. V., 2006. "Direct trajectory optimization and costate estimation via an orthogonal collocation method". *Journal of Guidance, Control, and Dynamics,* **29**(6), pp. 1435–1440.

[37] Basset, G., Xu, Y., and Li, N., 2013. "Fast trajectory planning via the b-spline augmented virtual motion camouflage approach". *Journal of Dynamic Systems, Measurement, and Control,* **135**(5), p. 054505.

[38] Wang, H., and Zou, Q., 2014. "B-spline-decomposition-based approach to multiaxis trajectory tracking: Nanomanipulation example". *IEEE Transactions on Control Systems Technology,* **22**(4), pp. 1573–1580.

[39] Murray, R. M., Li, Z., Sastry, S. S., and Sastry, S. S., 1994. *A mathematical introduction to robotic manipulation*. CRC press, Boca Raton.

[40] Dydek, Z. T., Annaswamy, A. M., and Lavretsky, E., 2013. "Adaptive control of quadrotor uavs: A design trade study with flight evaluations". *IEEE Transactions on Automation Science and Engineering*.

[41] Bouabdallah, S., Murrieri, P., and Siegwart, R., 2004. "Design and control of an indoor micro quadrotor". In Proceedings of the IEEE International Conference on

Robotics and Automation, Vol. 5, pp. 4393–4398.

[42] Fan, W., Gao, X.-S., Lee, C.-H., Zhang, K., and Zhang, Q., 2013. “Time-optimal interpolation for five-axis cnc machining along parametric tool path based on linear programming”. *The International Journal of Advanced Manufacturing Technology,* **69**(5-8), pp. 1373–1388.

[43] Zhou, J., Sun, Y., and Guo, D., 2014. “Adaptive feedrate interpolation with multiconstraints for five-axis parametric toolpath”. *The International Journal of Advanced Manufacturing Technology,* **71**(9-12), pp. 1873–1882.

[44] Heng, M. M.-T., 2008. “Smooth and time-optimal trajectory generation for high speed machine tools”. Master’s thesis, University of Waterloo.

[45] Wu, J., Zhou, H., Tang, X., and Chen, J., 2012. “A nurbs interpolation algorithm with continuous feedrate”. *International Journal of Advance Manufacture Technology,* **59**, pp. 623–632.

[46] Megiddo, N., 1984. “Linear programming in linear time when the dimension is fixed”. *Journal of the ACM (JACM),* **31**(1), pp. 114–127.

[47] Best, M. J., and Chakravarti, N., 1990. “Active set algorithms for isotonic regression; a unifying framework”. *Mathematical Programming,* **47**(1-3), pp. 425–439.

[48] Hom, R. A., and Johnson, C. R., 1991. “Topics in matrix analysis”. *Cambridge UP, New York.*