Waypoint-Based Generation of Guided and Optimal Trajectories for Autonomous Tracking Using a Quadrotor UAV

Ghulam FARID[1]*, Haris TAHIR HAMID[2], Shahid KARIM[3], Sohaib TAHIR[1]

[1] Department of Electrical Engineering, Comsats University Islamabad,
Sahiwal Campus, 57000, Pakistan
farid.anjum@yahoo.com (**Corresponding author*), sohaibchauhdary@hotmail.com

[2] Department of Electrical Engineering, FAST NUCES Faisalabad, 38000, Pakistan
me.haris@live.com

[3] School of Electronics and Information Engineering, Harbin Institute of Technology,
Harbin, 150001, China
shahidhit@yahoo.com

Abstract: Waypoint-based trajectory tracking control of a quadrotor UAV with robust performance attributes requires the generation of dynamically feasible and smooth trajectories. The smoothness of these trajectories relies on the adopted interpolation methods along with the constraints being considered (e.g. velocity, acceleration, and input constraints etc.). This paper briefly discusses the optimal trajectory generation and investigates Runge's phenomenon by presenting various feasible interpolation methods. Runge's issue may become severe due to sampling between waypoints when corridor constraints are considered with higher ordered polynomial interpolation. Trajectory generation requires the system dynamics to be differentially flat, therefore, a detailed proof is presented as well which shows clearly that quadrotor underactuated dynamics are differentially flat. For corridor constraints, higher order polynomials generate the trajectories near boundary walls, which may result in collision. This problem is solved using a scheme that guides these polynomials from one waypoint to another by incorporating a guiding term. The degree of smoothness and the shape of the desired path can be adjusted through the afore-mentioned guiding term. The proposed trajectory generation scheme can effectively reduce Runge's phenomenon and still offer finite and smooth curvatures at waypoints. Furthermore, the quadrotor model along with classical linear controller is used to demonstrate the smooth and faithful tracking of the optimal trajectories generated.

Keywords: Trajectory generation, Guided polynomials, Quadrotor UAV, Aerial robot, Linear controller.

1. Introduction

Unmanned aerial vehicles (UAVs) have extensively been investigated by the researchers over the past and recent decades. Specifically, a huge body of research has been directed towards quadrotors with huge contributions towards its design and control [4, 9, 13]. Currently, major research focus is on intelligent swarm control and aggressive maneuvering with quadrotor UAVs [7, 12, 15, 16]. To perform such tasks, generation of smooth and dynamically feasible trajectories is inevitable [6, 8, 17]. Smoothness can be defined as a function being continuous over time along with its continuous first-order derivative. Sometimes, a second derivative having continuity over time is also desirable for a specific dynamical system. Rough trajectories cause an increased burden on the robot dynamics and its control system by exciting undesirable vibrations. Any smooth function of time can be used to characterize a path and the shape of the path depends on the type of the chosen function. Additional constraints can also be applied to these functions for further smoothness and to attain the desired characteristics [3, 11].

Characteristically, trajectory is a time history that holds information regarding position, velocity, acceleration, and orientation of the robot. The user does not need to specify the complicated functions of time and space for describing the trajectories. This work is usually done by the robot online and trajectories are specified by the users in simple descriptions i.e., initial and final positions of the robot or waypoints. Furthermore, other spatial constraints like the elapsed time, velocity, and acceleration profiles are also considered equally by the robot in order to bring smoothness and desired attributes in the trajectory to be generated. Differential flatness is an important theory which helps in generating the dynamically feasible spatial trajectories subject to the proof that output states of the dynamical system are also differentially flat or simply flat [11, 12]. Flatness offers a way of decomposing the desired UAV trajectory into a chain of flat outputs and their derivatives. This further enables the calculation of controller commands to follow the computed trajectory [14].

https://doi.org/10.24846/v27i2y201810

One of the major contributions of this paper is the provision of a detailed proof regarding flatness of the quadrotor dynamical states which verifies the associated flatness property. Trajectory for a quadrotor UAV consists of a time history of its position in Cartesian space along with its yaw angle information [3]. It can be thought of as a sequence of *n* polynomials for *n+1* intermediate waypoints. The order of the polynomials depends on the number of constraints available in order to determine the unknown polynomial coefficients and vice versa. To perform aggressive maneuvers, each polynomial segment in a trajectory may have different controller parameters for different goals. This paper also presents the process of generating the optimal and constrained trajectories for quadrotor UAVs with fundamental focus on realization of various constraints (e.g., minimum jerk, minimum snap, and corridors). In proposed trajectory generation algorithm, the trajectories can be specified in form of intermediate waypoints along with initial and final goal points, i.e. these points actually refer to frames which hold both position and orientation information.

The other major contribution of this study is the investigation of Runge's phenomenon which becomes severe when higher-ordered polynomials are used for interpolation in order to attain an enhanced degree of smoothness. Runge's phenomenon is basically an issue of oscillations at the edges of the waypoints and this causes the trajectory to go significantly away from an ideal desired path. Specifically, when corridor constraints are considered, the generation of new virtual equi-spaced waypoints makes the aforementioned issue severe. To address this problem, this study proposes a scheme that uses a guiding term in order to steer the higher-ordered polynomials with respect to user-defined guiding weight factor. Apart from user definable waypoints, this scheme also enables the users to define guiding weight in order to additionally alter the shapes of the trajectories.

Moreover, generated optimal trajectories are tracked using a controller that is developed from the dynamical model of the quadrotor UAV. Linearized model of the quadrotor is used for the realization of tracking control and linearization of the model is performed around the hover state by assuming small angle approximation. The rest of the paper is organized as follows. Section 2 presents preliminaries on quadrotor modeling and control design which are subsequently used to track the generated trajectories, section 3 discusses the differential flatness and the detailed proof is derived, section 4 explains interpolation, optimal trajectory generation and guided trajectory generation algorithms, while section 5 presents the numerical simulations performed in MATLAB. Finally, conclusions close the paper in section 6.

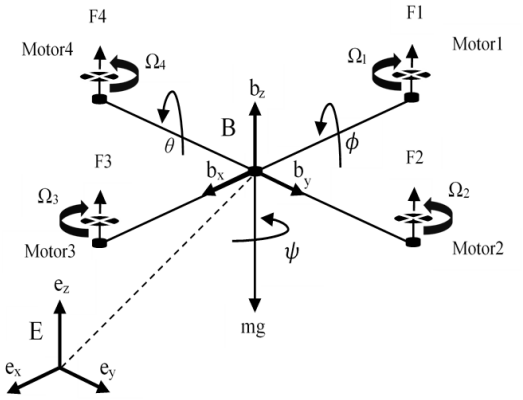

Figure 1. Quadrotor configuration with reference body and Earth frames

2. Preliminaries

2.1 Quadrotor Modeling

ξ_B and ω_B represent position and body angular rate of quadrotor model in body reference frame respectively. Both can be translated into earth reference frame by means of rotation matrix *R* and transfer matrix *T* correspondingly. Kinematics of quadrotor UAV can be expressed as:

$$\xi_E = R\xi_B \tag{1}$$

$$\dot{\Theta}_E = T\omega_B \tag{2}$$

where ξ_E, ξ_B represent quadrotor position vectors in earth and body reference frames respectively and can be characterized as $\xi = \begin{bmatrix} x & y & z \end{bmatrix}^T$ while $\omega_B = \begin{bmatrix} p & q & r \end{bmatrix}^T$ and $\dot{\Theta}_E = \begin{bmatrix} \dot{\varphi} & \dot{\theta} & \dot{\psi} \end{bmatrix}^T$ denote body angular rates and Euler angular rates respectively.

Orthogonal rotation matrix *R* and transfer matrix *T* are represented as:

$$R = \begin{bmatrix} \cos\theta\cos\psi & \sin\varphi\sin\theta\cos\psi - \cos\varphi\sin\psi & \sin\varphi\sin\psi + \cos\varphi\sin\theta\cos\psi \\ \cos\theta\sin\psi & \sin\varphi\sin\theta\sin\psi + \cos\varphi\cos\psi & \cos\varphi\sin\theta\sin\psi - \sin\varphi\cos\psi \\ -\sin\theta & \sin\varphi\cos\theta & \cos\varphi\cos\theta \end{bmatrix}$$

$$T = \begin{bmatrix} 1 & \sin\varphi\tan\theta & \cos\varphi\tan\theta \\ 0 & \cos\varphi & -\sin\varphi \\ 0 & \sin\varphi\sec\theta & \cos\varphi\sec\theta \end{bmatrix}$$

Matrix T depends on Euler angles and it is invertible if following conditions hold: for roll $\left(-\frac{\pi}{2}<\varphi<\frac{\pi}{2}\right)$, for pitch $\left(-\frac{\pi}{2}<\theta<\frac{\pi}{2}\right)$ and for yaw $(-\pi<\psi<\pi)$, otherwise singularity will occur [1]. Therefore, maneuvering beyond these constraints can't be performed using the Euler angles notion. Quaternion approach can be used to avoid these singularity issues [1, 2, 10].

Forces and moments that act on quadrotor model shown in Figure 1 can be written using Newton-Euler formalism as follows:

$$\begin{bmatrix} F_B \\ \tau_B \end{bmatrix} = \begin{bmatrix} mI_{3\times3} & 0 \\ 0 & I \end{bmatrix} \begin{bmatrix} \dot{V}_B \\ \dot{\omega}_B \end{bmatrix} + \begin{bmatrix} \omega_B \times mV_B \\ \omega_B \times I\omega_B \end{bmatrix} \tag{3}$$

Here I is the 3×3 inertia tensor and $I_{3\times3}$ is identity matrix, V represents linear speed vector of body and ω represents angular velocity vector. Position and velocity state equations can be written in inertial reference frame as:

$$\frac{d}{dt}\xi = V \tag{4}$$

$$\frac{d}{dt}V = -ge_z + \frac{F_B}{m}Re_z \tag{5}$$

$$\frac{d}{dt}R = RS(\omega_B) \tag{6}$$

where $S(\omega_B)$ is a skew symmetric matrix and

$$S(\omega_B) = \begin{bmatrix} 0 & -r & q \\ r & 0 & -p \\ -q & p & 0 \end{bmatrix}$$

There are two major dynamic entities produced by the rotation of the rotor, one is lift force or thrust and the other is the moment around the spinning axes. Thrust owns a key role in all types of quadrotor maneuvers, hence an important term in control design process. Different combinations of the propeller RPM constitute one thrust and three moment terms (i.e. control inputs) as follows:

$$\begin{cases} U_1 = b(\Omega_1^2+\Omega_2^2+\Omega_3^2+\Omega_4^2) \\ U_2 = b(\Omega_4^2-\Omega_2^2) \\ U_3 = b(\Omega_3^2-\Omega_1^2) \\ U_4 = d(\Omega_4^2+\Omega_2^2-\Omega_3^2-\Omega_1^2) \end{cases} \tag{7}$$

where U_1-U_4 denote control inputs, b and d represent thrust and drag factors respectively while Ω_i represents propeller speed.

By letting U_1 be equal to F_B and $\begin{bmatrix} lU_2 & lU_3 & U_4 \end{bmatrix}^T$ be equal to torque vector τ_B, we can extract following mathematical model in earth frame from (1-6).

$$\begin{cases} \ddot{x} = (\sin\varphi\sin\psi+\cos\varphi\sin\theta\cos\psi)\frac{1}{m}U_1 \\ \ddot{y} = (\cos\varphi\sin\theta\sin\psi-\sin\varphi\cos\psi)\frac{1}{m}U_1 \\ \ddot{z} = -g+(\cos\varphi\cos\theta)\frac{1}{m}U_1 \end{cases} \tag{8}$$

$$\begin{cases} \ddot{\varphi} = \dot{\theta}\dot{\psi}(\frac{I_y-I_z}{I_x}) - \frac{I_r}{I_x}\dot{\theta}\Omega_r + \frac{l}{I_x}U_2 \\ \ddot{\theta} = \dot{\varphi}\dot{\psi}(\frac{I_z-I_x}{I_y}) + \frac{I_r}{I_y}\dot{\varphi}\Omega_r + \frac{l}{I_y}U_3 \\ \ddot{\psi} = \dot{\varphi}\dot{\theta}(\frac{I_x-I_y}{I_z}) + \frac{1}{I_z}U_4 \end{cases} \tag{9}$$

2.2 Control Design

In this section, we compute the desired controller commands used to track the generated trajectories. Following equation can be derived from (8) in order to compute the desired thrust.

$$U_{1d} = m\left[\cos\phi\cos\theta\right]^T\left(K_p z_e + K_d \dot{z}_e + g\right) \tag{10}$$

where $z_e = z_d - z$ is error, K_p and K_d are tuning gains.

To derive the relation between the translational subsystem and rotational subsystem, linearization assumption is made around hover or quasi-stationary flight [5]. Thus, thrust input U_1 will be equal to mg for first two equations of (8), and letting roll and pitch angles be equal to zero, we have:

$$\ddot{x} = g(\Delta\varphi\sin\psi+\Delta\theta\cos\psi) \tag{11}$$

$$\ddot{y} = g(\Delta\theta\sin\psi-\Delta\varphi\cos\psi) \tag{12}$$

This generates:

$$\varphi_d = \frac{1}{g}\left(\ddot{x}\sin\psi - \ddot{y}\cos\psi\right) \tag{13}$$

$$\theta_d = \frac{1}{g}\left(\ddot{x}\cos\psi + \ddot{y}\sin\psi\right) \tag{14}$$

The equations $\ddot{x} = \ddot{x}_d + K_d\dot{e} + K_p e$ and $\ddot{y} = \ddot{y}_d + K_d\dot{e} + K_p e$ will force the system to follow desired Cartesian commands in *x*-*y* plane.

$$\begin{cases} U_{2d} = \frac{1}{l}(I_x(K_p\varphi_e + K_d\dot{\varphi}_e) + \dot{\theta}I_r\Omega_r - \dot{\theta}\dot{\psi}(I_y - I_z)) \\ U_{3d} = \frac{1}{l}(I_y(K_p\theta_e + K_d\dot{\theta}_e) - \dot{\varphi}I_r\Omega_r - \dot{\varphi}\dot{\psi}(I_z - I_x)) \\ U_{4d} = I_z(K_p\psi_e + K_d\dot{\psi}_e) - \dot{\varphi}\dot{\theta}(I_x - I_y) \end{cases} \tag{15}$$

where $\varphi_e = \varphi_d - \varphi$ and $\theta_e = \theta_d - \theta$.

Equation (7) can be used to define the relation between controller commands and rotor speeds as follows:

$$\begin{bmatrix} \Omega_{1d}^2 \\ \Omega_{2d}^2 \\ \Omega_{3d}^2 \\ \Omega_{4d}^2 \end{bmatrix} = \begin{bmatrix} b & b & b & b \\ 0 & -b & 0 & b \\ -b & 0 & b & 0 \\ -d & d & -d & d \end{bmatrix}^{-1} \begin{bmatrix} U_{1d} \\ U_{2d} \\ U_{3d} \\ U_{4d} \end{bmatrix} \tag{16}$$

where $\Omega_{id}^2, i = 1,2,3,4$ represent desired motor speed commands.

3. Differential Flatness

Differential flatness or simply the flatness offers a way of decomposing the desired UAV trajectory into a chain of flat outputs and their derivatives. This further enables the calculation of controller commands to follow the computed trajectory.

A system $\dot{x} = \delta(x,u)$ with input vector $u \in R^m$ and state vector $x \in R^n$ having a smooth vector function, is said to be differentially flat provided that a vector $y \in R^m$ exists in the following form:

$$y = \alpha\left(x, u, \dot{u}, \cdots u^{(r)}\right) \tag{17}$$

with following inverse smooth functions

$$x = \beta\left(y, \dot{y}, \cdots y^{(q)}\right) \tag{18}$$

$$u = \gamma\left(y, \dot{y}, \cdots y^{(q)}\right) \tag{19}$$

The flat outputs are defined by the vector $y = \alpha\left(x, u, \dot{u}, \cdots u^{(r)}\right)$. In other words, flatness is the process of expressing the selected outputs as a function of state vector x, control input vector u and its successive derivatives. From (17), it can be seen that the state vector and control input vector of a differentially flat system can be stated in terms of flat outputs and their derivatives.

This section shows that quadrotor underactuated dynamics are differentially flat and can be represented as a function of selected outputs along with their derivatives. Technically, this facilitates the expression of trajectory in terms of selected flat outputs along with the realization of constrained optimization by considering various boundary conditions on inputs and states of the system. The selected flat outputs of the quadrotor UAV are:

$$\chi = \left[x, y, z, \psi\right]^T \rightarrow R^3 \times SO(2) \tag{20}$$

A trajectory $\chi(t)$ can be defined as a time-variant smooth curve in the space of selected flat outputs of the system.

Theorem I:

The states of the quadrotor are the function of selected flat outputs and their derivatives.

Since the dynamics or states of the quadrotor UAV can be classified as translational and rotational types, to prove this theorem, we can divide the problem into two corresponding consequences.

Lemma 1:

Translational states are the function of selected flat outputs and their derivatives.

Proof:

We recall (8) and then write it in the following form:

$$m\ddot{\xi} = -mge_z + U_1 Rb_z \tag{21}$$

where $\ddot{\xi} = \begin{bmatrix} \ddot{x} & \ddot{y} & \ddot{z} \end{bmatrix}^T$.

By inspecting (21) it is obvious that translational dynamics reside in the second derivative of flat outputs.

The only thing that has to be proven is the dependence of *R* on the flat outputs. By considering m as a unit mass and $\|R\| = 1$, we have:

$$U_1 b_z = m\left(\ddot{\xi} + g e_z\right) \tag{22}$$

$$b_z = \frac{\left(\ddot{\xi} + g e_z\right)}{\left\|\ddot{\xi} + g e_z\right\|} \tag{23}$$

By defining a new unit vector b_n, which is orthogonal to b_z and makes an angle ψ with e_x in earth frame, we can determine b_y by using the following notion from [11]:

$$b_y = \frac{b_z \times b_n}{\left\|b_z \times b_n\right\|}, \quad b_x = b_y \times b_z \tag{24}$$

This gives

$$R = \begin{bmatrix} b_x & b_y & b_z \end{bmatrix} \tag{25}$$

Thus, (25) shows that *R* can be determined uniquely from the unit vectors in body frame. The unit vectors in body frame are the functions of translational states which further reside in the second derivative of flat outputs.

Lemma 2:

Rotational states (angular velocity and acceleration) of the body frame are the function of selected flat outputs and their derivatives.

Proof:

Considering (8) and applying small angle approximation, we have:

$$\begin{cases} \ddot{x} = (\sin\theta)\dfrac{1}{m}U_1 \\ \ddot{y} = (-\sin\varphi)\dfrac{1}{m}U_1 \\ \ddot{z} = -g + (\cos\varphi\cos\theta)\dfrac{1}{m}U_1 \end{cases} \tag{26}$$

The third derivative of (26) gives:

$$m x^{(3)} = (\sin\theta)\dot{U}_1 + (\dot{\theta}\cos\theta)U_1 \tag{27}$$

$$m y^{(3)} = -(\sin\varphi)\dot{U}_1 - (\dot{\varphi}\cos\varphi)U_1 \tag{28}$$

$$m z^{(3)} = (\cos\varphi\cos\theta)\dot{U}_1 - (\dot{\theta}\cos\varphi\sin\theta + \dot{\varphi}\cos\varphi\cos\theta)U_1 \tag{29}$$

Using (27, 28) and the relation $\begin{bmatrix} \dot{\varphi} & \dot{\theta} & \dot{\psi} \end{bmatrix}^T = T\begin{bmatrix} p & q & r \end{bmatrix}^T$ from (2), it is proved that the first two terms $\begin{bmatrix} p & q & - \end{bmatrix}^T$ of body angular velocity vector are a function of corresponding flat outputs. The third term of body angular rate cannot be determined using (29), however it can be extracted from (2) using the following relation:

$$\dot{\psi} = q\sin\varphi\sec\theta + r\cos\varphi\sec\theta \tag{30}$$

$$r = \frac{\dot{\psi} - q\sin\varphi\sec\theta}{\cos\varphi\sec\theta} \tag{31}$$

This verifies that angular velocity vector of the body frame is a function of flat outputs of the system.

The fourth derivative of (26) generates:

$$m x^{(4)} = (\sin\theta)\ddot{U}_1 + 2(\dot{\theta}\cos\theta)\dot{U}_1 - \left(\dot{\theta}^2\sin\theta - \ddot{\theta}\cos\theta\right)U_1 \tag{32}$$

$$m y^{(4)} = -(\sin\varphi)\ddot{U}_1 - 2(\dot{\varphi}\cos\varphi)\dot{U}_1 + \left(\dot{\varphi}^2\sin\varphi - \ddot{\varphi}\cos\varphi\right)U_1 \tag{33}$$

$$\begin{aligned} m z^{(4)} = {} & -2(\dot{\theta}\cos\varphi\sin\theta + \dot{\varphi}\sin\varphi\cos\theta)\dot{U}_1 + \\ & 2\left(\dot{\varphi}\dot{\theta}\sin\varphi\sin\theta\right)U_1 - \\ & \left(\dot{\varphi}^2 + \dot{\theta}^2\right)U_1\cos\varphi\cos\theta + \left(\cos\varphi\cos\theta\right)\ddot{U}_1 - \\ & \left(\ddot{\varphi}\sin\varphi\cos\theta + \ddot{\theta}\cos\varphi\sin\theta\right)U_1 \end{aligned} \tag{34}$$

It is obvious from (32, 33) that first two terms of body angular acceleration can be defined explicitly in the flat output space of the system while the remaining third term can be determined by following the same procedure as the one adopted to obtain (31).

Theorem II:

The inputs of the quadrotor are the function of selected flat outputs and their derivatives.

Proof:

Simplifying the rotational dynamics of quadrotor from (9), we have:

$$\begin{cases} \ddot{\varphi} = \dfrac{l}{I_x} U_2 \\ \ddot{\theta} = \dfrac{l}{I_y} U_3 \\ \ddot{\psi} = \dfrac{1}{I_z} U_4 \end{cases} \tag{35}$$

For brevity, normalizing the inertial and length terms in (35), we get a direct relation between the inputs and angular acceleration states. From (32-34) and (35), we can write:

$$m\begin{bmatrix} x^{(4)} \\ y^{(4)} \\ z^{(4)} \end{bmatrix} = \begin{bmatrix} 2(\dot{\theta}\cos\theta)\dot{U}_1 - \left(\dot{\theta}^2 \sin\theta\right)U_1 \\ -2(\dot{\varphi}\cos\varphi)\dot{U}_1 + \left(\dot{\varphi}^2 \sin\varphi\right)U_1 \\ -2(\dot{\theta}\cos\varphi\sin\theta + \dot{\varphi}\sin\varphi\cos\theta)\dot{U}_1 + \\ 2\left(\dot{\varphi}\dot{\theta}\sin\varphi\sin\theta\right)U_1 - \left(\dot{\varphi}^2 + \dot{\theta}^2\right)U_1\cos\varphi\cos\theta \end{bmatrix}$$
$$+\begin{bmatrix} \sin\theta & 0 & (\cos\theta)U_1 \\ -\sin\varphi & -(\cos\varphi)U_1 & 0 \\ \cos\varphi\cos\theta & -(\sin\varphi\cos\theta)U_1 & -(\cos\varphi\sin\theta)U_1 \end{bmatrix}\begin{bmatrix} \ddot{U}_1 \\ U_2 \\ U_3 \end{bmatrix} \tag{36}$$

and

$$\ddot{\psi} = \frac{1}{I_z} U_4 \tag{37}$$

Thus, from (36, 37), it is obvious that all the control inputs are the function of flat outputs and their derivatives. The inputs U_2 and U_3 occur in the 4th derivative of the translational dynamics of the quadrotor while U_4 appears in the second derivative of the yaw angle. The theorems presented above prove that the quadrotor is a differentially flat system and can be expressed as a function of flat outputs using a chain of integrators.

4. Interpolation and Trajectory Generation

In the introductive section, it has already been discussed that initial and final goal points are not merely the points, but rather the frames which hold information regarding position in space along with the yaw angle of a quadrotor UAV. Similarly, each intermediate point can be considered as a frame and we can assume there are $n+1$ frames (including start and end points). Therefore, we can formulate the navigation problem by considering that the quadrotor has to navigate through $n+1$ frames. Trajectories are usually higher dimensional problems and solution is computed by splitting the problem into multiple single-dimensions. A simple way of generating the trajectory between the frames is by straight line interpolation. The problem is that this causes aggressive curves at the end of each intermediate frame and quadrotor needs to stop there which obviously is not a good choice. However, polynomials of different orders can be used to interpolate smoothly between these waypoints.

A polynomial can be written in the following form:

$$\begin{aligned} & p_i(t) = c_{i0} + c_{i1}t + c_{i2}t^2 + \cdots + c_{im}t^m \\ & \forall \quad i = 0 \cdots n \end{aligned} \tag{38}$$

$$p_i(t) = \sum_{j=0}^{m} c_{ij} t^j \tag{39}$$

Using (38), a complete trajectory can be formulated over various time slots as follows:

$$p(t) = \begin{cases} \sum_{j=0}^{m} c_{0j} t^j & \\ \sum_{j=0}^{m} c_{1j} t^j & \text{for} \quad t_0 \le t < t_1 \\ \vdots & t_1 \le t < t_2 \\ \sum_{j=0}^{m} c_{nj} t^j & \vdots \\ & t_n \le t < t_{n+1} \end{cases} \tag{40}$$

where m defines the degree of polynomial.

We can further write (38) as:

$$p_i(t) = \left[1, t, t^2, \cdots, t^m\right] C \tag{41}$$

$$p(t) = T(t) C \tag{42}$$

where C is the coefficients' vector, and $T(t) = \left[1, t, t^2, \cdots, t^m\right]$ characterizes the polynomial of position.

Taking the successive four derivatives of T yields velocity, acceleration, jerk and snap respectively as follows:

$$\begin{cases} \dot{T}(t) = \left[0,1,2t,3t^2,\cdots,mt^{(m-1)}\right] \\ \ddot{T}(t) = \left[0,0,2,6t,12t^2,\cdots,m(m-1)t^{(m-2)}\right] \\ T^{(3)}(t) = \left[0,0,0,6,24t,\cdots,m(m-1)(m-2)t^{(m-3)}\right] \\ T^{(4)}(t) = \left[0,0,0,0,6,24,\cdots,m(m-1)(m-2)(m-3)t^{(m-4)}\right] \end{cases} \tag{43}$$

4.1 Generalized Constraints

In order to determine the coefficient vector C, we need to consider some generalized constraints. A multi-dimensional problem is always solved by splitting it up into single-phase problems and then the procedure is repeated separately for each dimension to construct a higher-dimensional trajectory. Therefore, all the constraints considered here imply single dimensional problems.

Constraint 1:

For a specific trajectory having $n+1$ waypoints, there exist n polynomials and each polynomial lies between pairs of known positions of waypoints. Therefore, the polynomials should pass through all the waypoints at different allocated time slots.

$$p_i(0) = w_i = c_{ij} \qquad \text{for} \quad i,j = 0 \tag{44a}$$

$$p_i(\tau_i) = w_{i+1} \qquad \text{for} \quad i = 1\cdots n \tag{44b}$$

where τ_i characterizes the time slots for different polynomials of the trajectory.

Mathematically τ_i can be written as $\tau_i = \sum_{k=0}^{i-1} t_k$ and generalized time variable in (38) can be replaced by $t = \dfrac{t-\tau_i}{t_i}$ in order to consider the real-time problem.

Constraint 2:

There always exist known start and stop positions in terms of defined waypoints and quadrotor must be at rest at these initial and final goal points.

$$p_i^{(k)}(0) = 0 \begin{cases} for & m=3 & 0 \le k \le 1 \\ for & m=5 & 0 \le k \le 2 \\ for & m=7 & 0 \le k \le 3 \end{cases} \tag{45a}$$

$$p_i^{(k)}(\tau_n) = 0 \begin{cases} for & m=3 & 0 \le k \le 1 \\ for & m=5 & 0 \le k \le 2 \\ for & m=7 & 0 \le k \le 3 \end{cases} \tag{45b}$$

Here, it is important to note that m defines the order of polynomial and k defines the derivative index.

Constraint 3:

In order to have a smooth transition at waypoints, the successive derivatives of the polynomials must be equal at these points. The order of the polynomial defines the index of the derivatives that would have to be taken for a smooth connection of the polynomials.

$$p_{i-1}^{(k)}(\tau_i) = p_i^{(k)}(\tau_i) \qquad \text{for} \quad i = 1\cdots n, \quad \text{and}$$

$$\begin{cases} for & m=3 & 1 \le k \le 2 \\ for & m=5 & 1 \le k \le 3 \\ for & m=7 & 1 \le k \le 4 \end{cases} \tag{46}$$

Considering all the above-mentioned constraints, we can formulate our problem as:

$$T_{n(m+1)\times n(m+1)} \cdot \mathrm{C}_{n(m+1)\times 1} = w_{n(m+1)\times 1} \tag{47a}$$

and the coefficient matrix can be solved as:

$$\mathrm{C}_{n(m+1)\times 1} = T^{-1}_{n(m+1)\times n(m+1)} \cdot w_{n(m+1)\times 1} \tag{47b}$$

4.2 Optimal Trajectory Generation

Actually, trajectories are often two-, three- or even higher dimensional and each dimension is usually solved separately. Optimal trajectory algorithm may incorporate various constraints such as sensor saturation constraints, moment and thrust constraints, corridor constraints, time, velocity, acceleration and jerk constraints, and minimization of various functionals, e.g. minimization of acceleration, jerk or snap etc. We can recall (42) to parameterize the constrained optimization problem as follows:

$$p_i(t) = T_i(t) C_i \qquad \text{for} \quad i = x,y,z,\psi \tag{48}$$

$$\min \int_{t_0}^{t_m} \left\| p_i^{(n_i)}(t) \right\|^2 .dt \,,$$

$$\begin{cases} for & m=3, & n_{x,y,z}=2, n_\psi = 2 \\ for & m=5, & n_{x,y,z}=3, n_\psi = 2 \\ for & m=7, & n_{x,y,z}=4, n_\psi = 2 \end{cases} \tag{49a}$$

$$\min \int_{t_0}^{t_m} \left\| T_i^{(n_i)}(t) C_i \right\|^2 .dt \tag{49b}$$

$$\min \quad C_i^T \left[\int_{t_0}^{t_m} \left[T_i^{(n_i)}(t) \right]^T \left[T_i^{(n_i)}(t) \right] .dt \right] C_i \tag{49c}$$

$$\min \quad C_i^T Q_i C_i \text{ , s.t. } A_{eq} C = B_{eq} \text{ , and}$$

$$A_{ieq} C \le B_{ieq} \tag{49d}$$

Here, in (49a) the objective function may be used for minimization of any functional e.g. the minimization of acceleration, jerk or snap. The optimal problem formulated above is solved using the Optimization Toolbox from MATLAB and the numerical solver used in this study is *quadprog ()*. However, some other numerical solvers, e.g. OOQP, CPLEX, Gurobi, etc., can also be used as optimizers.

4.3 Corridor Constraints

Typically, minimizing the functionals which involve higher order polynomials generates the trajectories that reside substantially away from the straight path. In order to limit the swing of the trajectories, the concept of corridor is usually incorporated in the optimization problem. Technically, a corridor is a feasible channel for which the trajectory must be inner-bounded. The intuitive train of thought is that if the corridor can be added as a constraint to the QP problem, the solution trajectory will naturally be in the corridor. To add corridor constraints, the sampling of the intermediate path between each waypoint is usually carried out and corridor constraints are considered at each sampled point of the trajectory. Following corridor inequality constraints can be incorporated into the optimization problem:

$$\begin{cases} x_{\min} \le p_s t_x \le x_{\max} \\ y_{\min} \le p_s t_y \le y_{\max} \\ z_{\min} \le p_s t_z \le z_{\max} \end{cases} \tag{50}$$

where $p_s t_{x,y,z}$ characterize the sampling points and each point is constrained by user-defined values of corridor coordinates.

4.4 Guided Trajectory

The trajectory that is generated using corridor constraints can still be shaped using guided polynomials. In some specific applications, where it is necessary to keep the UAV substantially away from the corridor boundaries, polynomials can be guided and the guiding weight can be adjusted depending on the requirement of the degree of smoothness of the path. As discussed, sampling of the intermediate path between each pair of waypoints is inevitable in order to add corridor constraints at these sampling points, therefore, these sampling points are the new waypoints or the virtual waypoints and the polynomials connecting these virtual waypoints can be thought of as virtual polynomials. The number of these virtual polynomials purely depends on number of sampling intervals. These virtual polynomials can be guided using modified constraints which involve steering linear-curve coefficients and consequently these constraints can be solved using optimization solvers (i.e. *quadprog*).

Suppose λ_i and λ_{i+1} are two consecutive virtual points at t_i and t_{i+1} respectively.

$$v_{i+1}^i = \frac{\lambda_i - \lambda_{i+1}}{t_i - t_{i+1}} \qquad \text{for} \quad i = 1, \cdots, n_v \tag{51}$$

$$\lambda_{i-1} = \lambda_i - v_{i+1}^i t_i \tag{52}$$

Using (38), any order of polynomial between these virtual points can be used and can be guided using the following formulation:

$$C_{guided} = \beta p_i(t) \left[\lambda_{i-1} \quad v_{i+1}^i \cdots zeros \right]^T \tag{53}$$

where β symbolizes the guiding coefficient and the resulting constraint can be considered in optimization problem.

Guided polynomials solve Runge's phenomenon which occurs due to the usage of higher order polynomial interpolation specifically when the corridor is considered and the path between adjacent waypoints is sampled. Trajectories reside far away from the ideal realization of the path due to this phenomenon and may result in a collision with other UAVs in the presence of any turbulence or nonlinear actuation. This intuitive scheme offers a simplistic way to shape the trajectories by altering only guiding weight β. In typical planning problems, this can be set manually by the user during the planning of waypoints to be followed.

5. Simulations

In this section, results of various numerical simulations are presented. Figure 2 shows the single dimensional trajectories which interpolate between various waypoints through polynomials

of different orders. The linear interpolation is computationally fast and simple; however, it suffers from infinite curvatures at waypoints and a smooth transition is not possible. The rest of the curves use 3rd-, 5th- and 7th-order polynomials and the swing of the trajectories increases as we move towards higher order polynomials. This ultimately brings in a severe form of Runge's phenomenon and obviously points out that usage of higher order polynomials does not always bring smoothness, instead, this increases the burden on the quadrotor's control ability. Figure 3 depicts various interpolation methods along with velocity and acceleration profiles. The important method depicted is the linear interpolation with parabolic blends, which can give a computationally fast way of trajectory generation. Also, various types of velocity and acceleration profiles are shown which gives a deep insight into how the interpolation behaves between two waypoints. The generalized constraints and optimal formulation presented in section 4, are considered to generate the two-dimensional optimal trajectory for quadrotor UAV as it is shown in Figure 4. This Figure illustrates that a two-dimensional problem can be solved by splitting it up into two separate single dimensional problems and then by solving each problem distinctly. The same procedure can be augmented for the solution of higher dimensional trajectories. Figure 5 illustrates position, velocity, acceleration and jerk profiles of the optimal minimum jerk trajectory. Similarly, Figure 6 shows various associated profiles of a minimum snap trajectory. Both these trajectories (i.e. the minimum jerk, and the minimum snap one) are constructed using the formulation developed in section 4. Figure 7 illustrates the minimum snap trajectory based on corridor constraints and it clearly shows that such trajectories, which rely on higher order polynomial interpolation, may collide with the walls of the corridor in case that any malfunctioning of actuators or severe wind turbulence occurs. The solution to this problem is proposed and illustrated in Figure 8, where each trajectory depends on guiding weight and can be shaped by varying this weighting factor. Moreover, this scheme also eradicates Runge's phenomenon if higher order polynomials are used for interpolation. Finally, Figures 9 and 10 show the generation and tracking of minimum snap trajectory by a quadrotor model using a linearized controller which has been developed in section 2. Initially, various waypoints are defined then the trajectory generation algorithm generates the optimal trajectory at runtime and the controller tracks this trajectory faithfully.

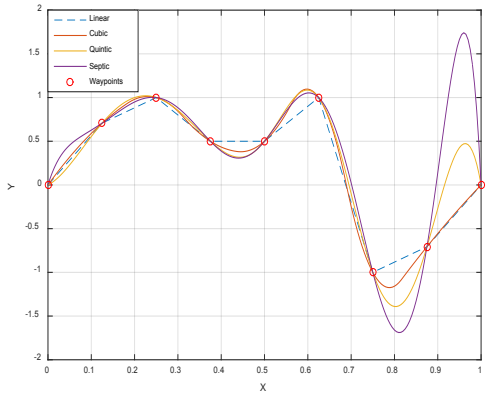

Figure 2. Interpolation using polynomials of different orders (the 7th-order polynomial curve is far away from the ideal realization of path, which ultimately leads towards Runge's phenomenon and brings oscillations)

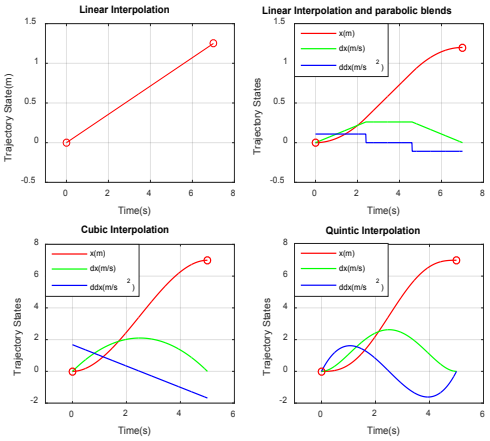

Figure 3. Different Interpolation methods along with velocity and acceleration profiles for a single segment

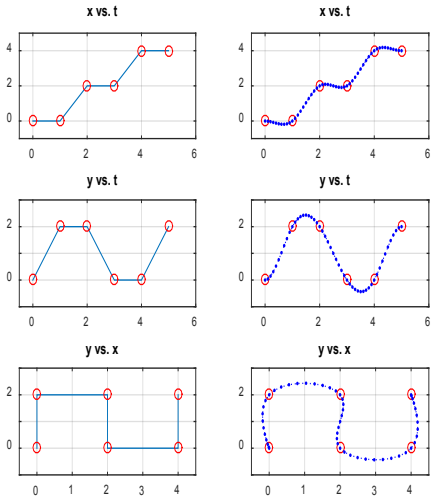

Figure 4. Two-dimensional trajectory generation method for multiple waypoints

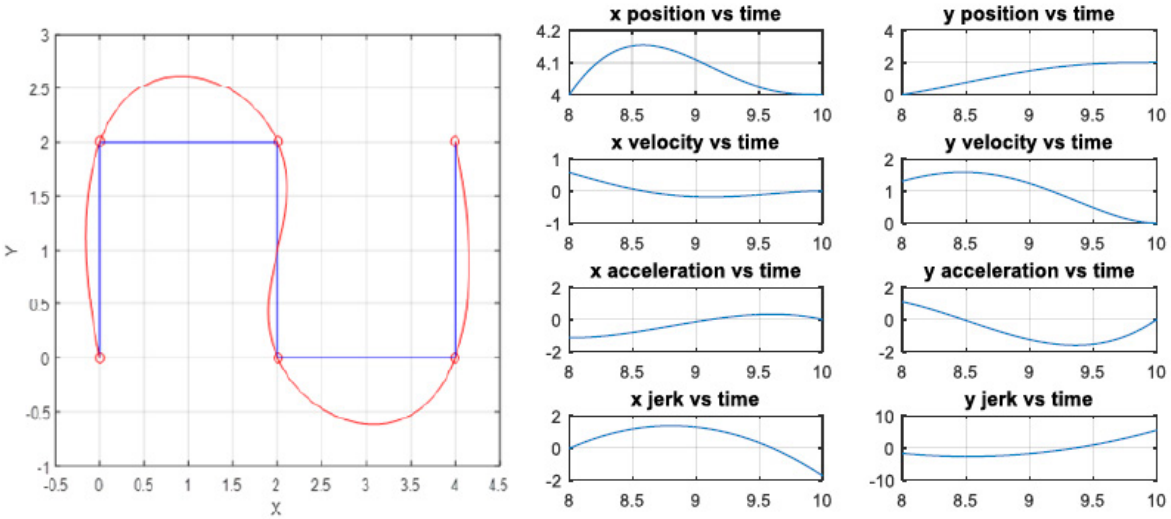

Figure 5. Optimal trajectory using 5th order polynomials (minimum jerk trajectory)

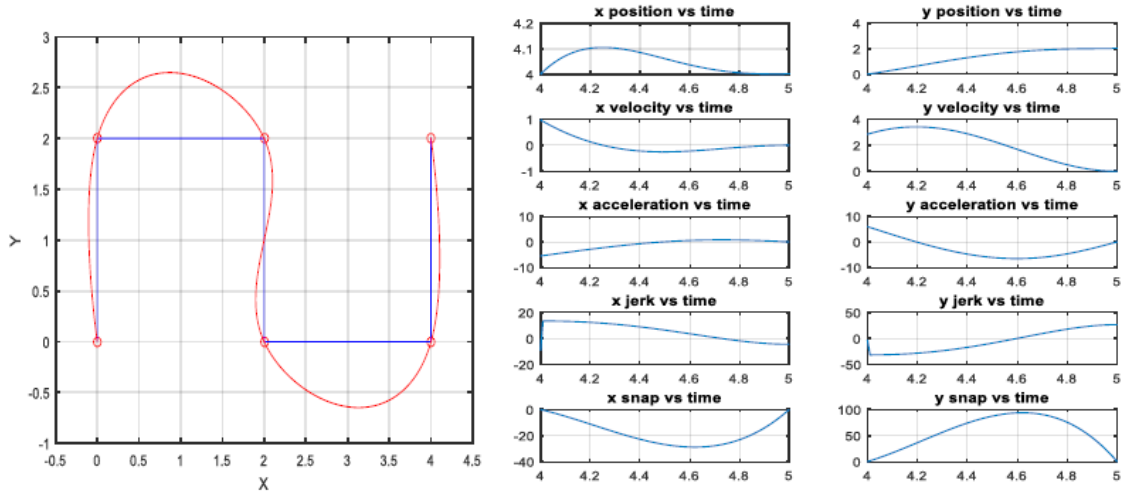

Figure 6. Optimal trajectory using 7th order polynomials (minimum snap trajectory)

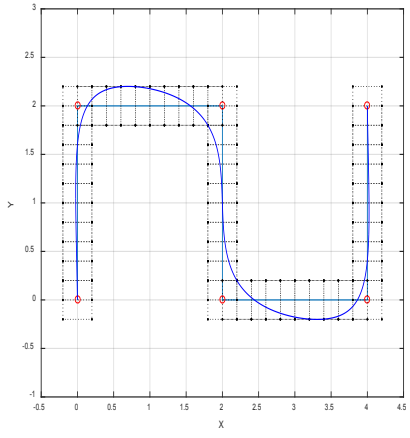

Figure 7. Optimal trajectory generation with corridor constraints

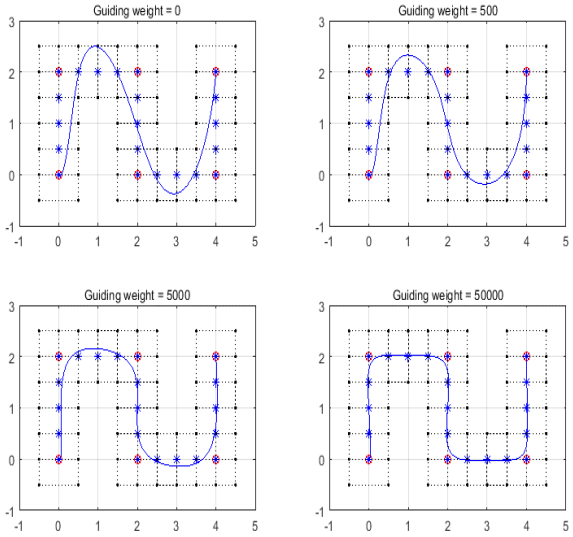

Figure 8. Trajectory generation using guided polynomials

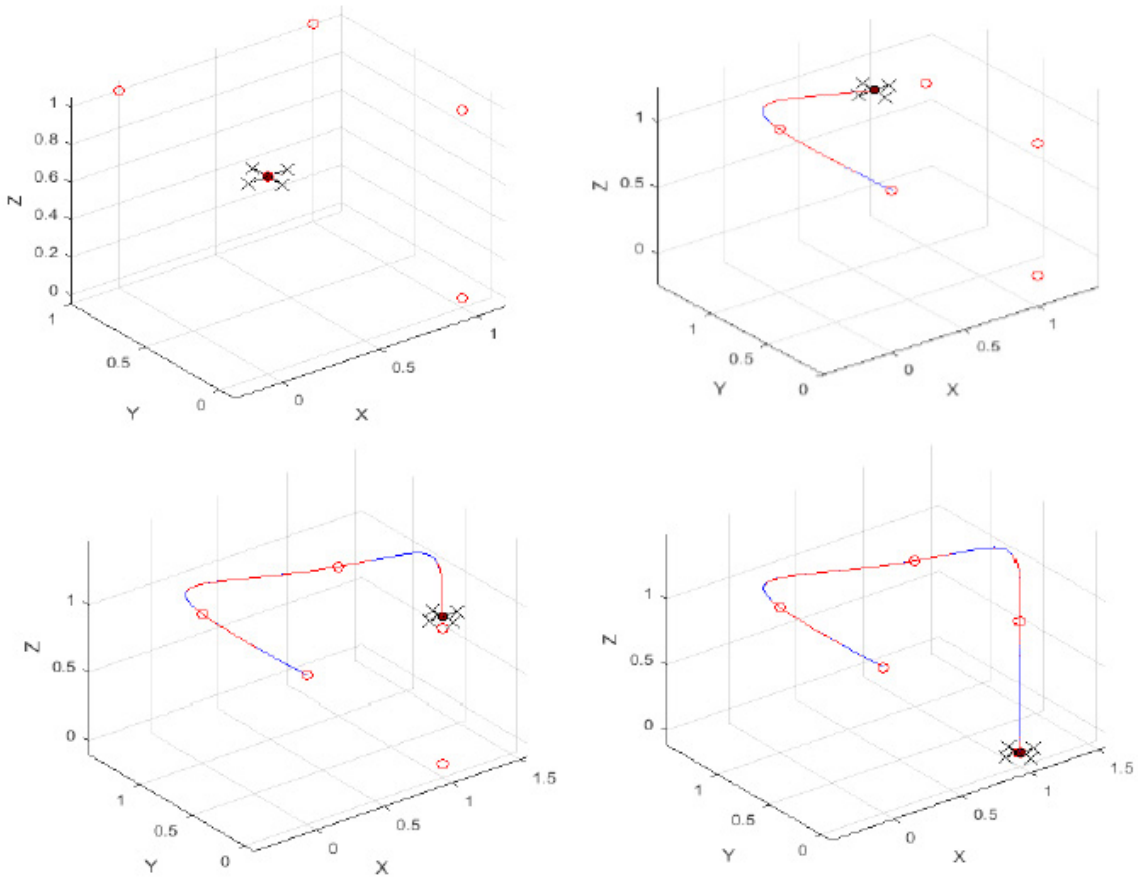

Figure 9. Waypoint-based trajectory generation and tracking using derived controller (minimum snap trajectory)

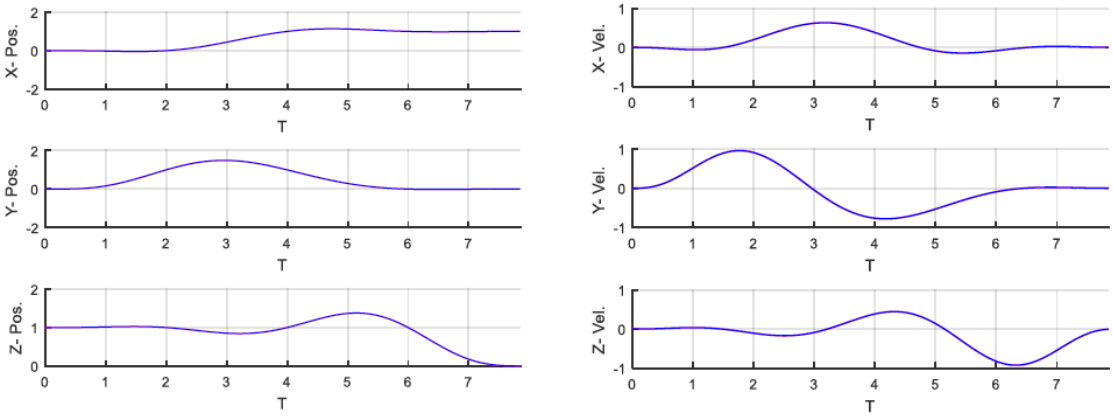

Figure 10. Position and velocity profiles of the tracked trajectory in Figure 9

6. Conclusions

In this study, a comprehensive formulation for generating various optimal and constrained trajectories for a quadrotor UAV has been presented. Firstly, we put forward a detailed proof of differential flatness which essentially allows a way to describe the trajectories in flat output space. Secondly, methods of planning the optimal trajectories using polynomials of various orders, are presented. Simulation analysis of different interpolation schemes has shown that Runge's phenomenon becomes severe as the polynomial order increases and therefore, the generated trajectories increase a significant burden on the UAV dynamics and controller performance. A method that uses guided polynomials is adopted to shape the trajectories and the afore-mentioned problem is avoided. In addition, this scheme restricts the UAV trajectory to stay substantially

away from the corridor walls so that any collision is avoided. Lastly, the waypoint-based optimal trajectory is faithfully followed by the quadrotor UAV using the derived controller equations.

REFERENCES

1. Arellano-Muro, C. A., Castillo-Toledo, B., Loukianov, A. G., Luque-Vega, L. F. & González-Jiménez, L. E. (2015). Quaternion-based trajectory tracking robust control for a quadrotor. In *System of Systems Engineering Conference (SoSE)*, San Antonio, TX, USA (pp. 386-391).

2. Chovancová, A., Fico, T., Hubinský, P. & Duchoň, F. (2016). Comparison of various quaternion-based control methods applied to quadrotor with disturbance observer and position estimator, *Robotics and Autonomous Systems*, *79*(1), 87-98.

3. Castillo-García, P., Muñoz Hernandez, L. E. & García Gil, P. (2017). Chapter 9 - Trajectory Generation, Planning & Tracking*, *Indoor Navigation Strategies for Aerial Autonomous Systems*, 213-241. Butterworth-Heinemann.

4. Farid, G., Mo, H., Ali, S. M. & Liwei, Q. (2017). A review on linear and nonlinear control techniques for position and attitude control of a quadrotor, *Control and Intelligent Systems*, *45*(1), 43-57.

5. Farid, G., Mo, H., Ahmed, M. I. & Ehsan, A. (2018). On control law partitioning for nonlinear control of a quadrotor UAV. In *2018 15th International Bhurban Conference on Applied Sciences and Technology (IBCAST)*, Islamabad, Pakistan (pp. 257 - 262).

6. Garcia, G. A. & Keshmiri, S. S. (2016). Biologically inspired trajectory generation for swarming UAVs using topological distances, *Aerospace Science and Technology*, *54*(Supplement C), 312-319.

7. Han, T., Chi, M., Guan, Z.-H., Hu, B., Xiao, J.-W. & Huang, Y. (2017). Distributed Three-Dimensional Formation Containment Control of Multiple Unmanned Aerial Vehicle Systems, *Asian Journal of Control*, *19*(3), 1103-1113.

8. Hehn, M. & D'Andrea, R. (2011). Quadrocopter Trajectory Generation and Control. In *IFAC Proceedings Volumes*, *44*(1) (pp. 1485-1491).

9. Kendoul, F. (2012). Survey of advances in guidance, navigation, and control of unmanned rotorcraft systems, *Journal of Field Robotics*, *29*(2), 315-378.

10. Liu, H., Wang, X. & Zhong, Y. (2015). Quaternion-Based Robust Attitude Control for Uncertain Robotic Quadrotors, *IEEE Transactions on Industrial Informatics*, *11*(2), 406-415.

11. Mellinger, D. & Kumar, V. (2011). Minimum snap trajectory generation and control for quadrotors. In *2011 IEEE International Conference on Robotics and Automation (ICRA)*, Shanghai, China (pp. 2520-2525).

12. Mellinger, D., Michael, N. & Kumar, V. (2012). Trajectory generation and control for precise aggressive manoeuvres with quadrotors, *The International Journal of Robotics Research*, *31*(5), 664-674.

13. Mo, H. & Farid, G. (2019). Nonlinear and adaptive intelligent control techniques for quadrotor UAV- a survey, *Asian Journal of Control*, *21*(3), 1-20.

14. Nguyen, N. T., Prodan, I., Stoican, F. & Lefèvre, L. (2017). Reliable nonlinear control for quadcopter trajectory tracking through differential flatness, *IFAC-Papers Online*, *50*(1) (pp. 6971-6976).

15. Pretorius, A. & Boje, E. (2014). Design and Modelling of a Quadrotor Helicopter with Variable Pitch Rotors for Aggressive Manoeuvres, *IFAC Proceedings Volumes*, *47*(3) (pp. 12208-12213).

16. Roldão, V., Cunha, R., Cabecinhas, D., Silvestre, C. & Oliveira, P. (2014). A leader-following trajectory generator with application to quadrotor formation flight, *Robotics and Autonomous Systems*, *62*(10), 1597-1609.

17. Yang, Z., Fang, Z. & Li, P. (2016). Bio-inspired Collision-free 4D Trajectory Generation for UAVs Using Tau Strategy, *Journal of Bionic Engineering*, *13*(1),84-97.