2021 International Conference on Unmanned Aircraft Systems (ICUAS)
Athens, Greece. June 15-18, 2021

Discrete Time Optimal Trajectory Generation and Transversality Condition with Free Final Time

Hossein Eslamiat *, Amit K. Sanyal †, Clark Lindsay ‡

A discrete time, optimal trajectory planning scheme for position trajectory generation of a vehicle is given here, considering the mission duration as a free variable. The vehicle is actuated in three rotational degrees of freedom and one translational degree of freedom. This model is applicable to vehicles that have a body-fixed thrust vector direction for translational motion control, including fixed-wing and rotorcraft unmanned aerial vehicles (UAVs), unmanned underwater vehicles (UUVs) and spacecraft. The lightweight scheme proposed here generates the trajectory in inertial coordinates, and is intended for real time, on-the-go applications. The unspecified terminal time can be considered as an additional design parameter. This is done by deriving the optimality conditions in a discrete time setting, which results in the discrete transversality condition. The trajectory starts from an initial position and reaches a desired final position in an unspecified final time that ensures the cost on state and control is optimized. The trajectory generated by this scheme can be considered as the desired trajectory for a tracking control scheme. Numerical simulation results validate the performance of this trajectory generation scheme used in conjunction with a nonlinear tracking control scheme.

I. Introduction

This paper considers autonomous online trajectory planning for a specific class of maneuvering vehicles operating in three-dimensional Euclidean space. It is especially vital to have on-board real-time trajectory planning in cases where the autonomously operated vehicle must operate in beyond-visual-line-of-sight (BVLOS) conditions and for operations in cluttered and dynamic environments that are either uncharted or poorly known to the vehicle a priori [1]. Indoor vehicle operations [2], package delivery in urban and suburban areas, monitoring of civilian infrastructures like bridges and highways, autonomous landing on moving platforms [3], and tracking wildlife in forested areas all represent applications where the model presented here can be used. In this paper we consider the problem of planning and producing a time trajectory for a vehicle's position in a three-dimensional Euclidean space, using a given initial waypoint and a desired final waypoint in position, and operating over an unspecified time. The position trajectory is then generated from the two given waypoints that are prescribed in terms of their position vectors. This trajectory generation scheme is applied to a vehicle that is underactuated, with four independent control inputs for the six degrees of freedom. The control inputs actuate three degrees of rotational motion and one degree of translational motion in a vehicle body-fixed coordinate frame. Control of the translational motion is achieved using a single thrust force along a body-fixed direction vector, while the direction of this body-fixed thrust vector is controlled by regulating the attitude (orientation) of the vehicle. This actuation model covers a broad spectrum of unmanned vehicles and can be applied to fixed-wing and quadcopter unmanned aerial vehicles (UAVs), unmanned underwater vehicles and spacecraft.

Past research on trajectory planning can be classified into four types. The first type either decouples time and geometry, constructs a geometric trajectory, and then parameterizes it in time, e.g. [4, 5], or uses Bezier curves to create trajectories [6]. The second type utilizes differential flatness of dynamics to generate a trajectory [7]. The third type uses a high-level trajectory generator in conjunction with a motion primitive generator to choose an optimized trajectory among different motion primitives [8]. The fourth and final type utilizes artificial neural networks to generate a trajectory. Examples of this final type are [9, 10] in which we used reinforcement learning to model the relationship between the local environment and the vehicle's dynamics in order to plan obstacle-free and smooth trajectories. These algorithms use preassigned flight times between adjacent waypoints, which means flight times are not considered design parameters. We improve those algorithms in this paper by considering flight time as an unspecified parameter that can be optimized to minimize a performance index. Another advantage of the proposed algorithm is its fast response to changes in waypoint locations. As new waypoints are introduced mid-flight, a new trajectory can be generated accordingly. Another recent work [11] plans a path that optimizes certain criteria such as fuel efficiency, and is able to handle a wide range of optimal maneuvers given arbitrary initial and final states relative to a cost function. The method

*Corresponding Author; Assistant Professor, Mechanical Engineering, Southern Illinois University, hossein.eslamiat@siu.edu
†Associate Professor, Mechanical and Aerospace Engineering, Syracuse University, aksanyal@syr.edu
‡Student, Mechanical Engineering and Computer Science, Southern Illinois University, clark.lindsay@siu.edu

2021 International Conference on Unmanned Aircraft Systems (ICUAS) | 978-1-6654-1535-4/21/$31.00 ©2021 IEEE | DOI: 10.1109/ICUAS51884.2021.9476719

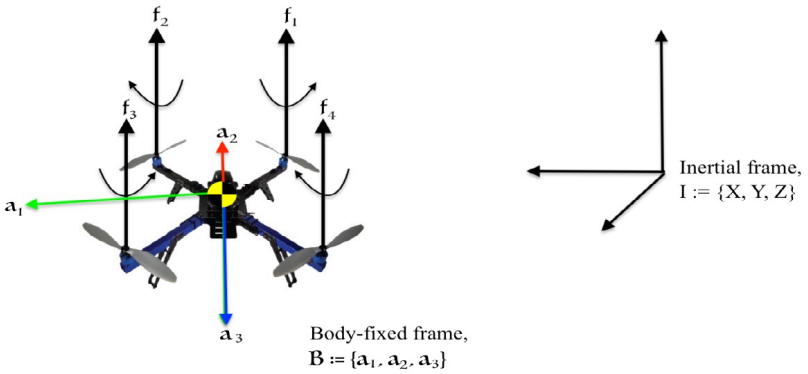


Fig. 1 Rigid body model considered for trajectory generation

in [12], similar to ours, uses waypoints to generate a flight trajectory, with focus on reducing computation time by using algebraic methods to avoid poor local minima and achieve the lowest time consumption possible. Trajectory planning can also be considered as a smoothing problem [13]; however, this problem is ill-posed [14] and in addition, requires having all the waypoints at once to create a trajectory, whereas, with on-board applications, the vehicle might need to generate segments of a trajectory one at a time. Trajectory planning schemes for quadrotor UAVs that satisfy the body-fixed thrust vector actuation model previously discussed have been treated in the past, for example, in [15]. For large and rapid maneuvers that go beyond hovering or level flight, control can be achieved using asymptotically stable or finite-time stable attitude control schemes with a large domain of convergence, as in [16]. Large maneuvers may also be necessary for obstacle and collision avoidance, as well as to recover from external disturbances that induce large rotations in the vehicle. In such large maneuver situations, fast online trajectory planning schemes like the one outlined here become necessary. An example of a previously researched method that allows for quickly re-planning trajectories is [17]. The method presented here will similarly allow for rapid trajectory correction and will be able to effectively take into account new waypoints.

In this paper trajectory generation between an initial state and a final state is approached as a discrete-time optimal control problem where the final time is free. The two main contributions of this paper are: (1) deriving the transversality condition in a discrete time setting that enables us to consider the final time as an additional design parameter; and (2) generating an optimal, online trajectory between the initial and final states for a tracking controller to follow. To the best of our knowledge, the trajectory planning scheme proposed here that derives and solves the transversality condition (for unspecified flight time) in a discrete time setting, has not been carried out in the past. The contents of this paper are organized as follows. Section II provides the dynamics model for the underactuated vehicle, with particular emphasis on the translational dynamics controlled by a single thrust along a body-fixed direction. Both the continuous-time dynamics and its discretization in time are presented in this section. In section III, the optimal position trajectory planning problem is posed and solved in discrete time, and the optimality condition corresponding to free terminal time step is presented. This condition is the well-known *transversality condition* [18], but derived here in discrete time. Simulations are presented in Section IV, where we consider the full six degrees of freedom tracking control of a quadcopter UAV, and show the application of the proposed scheme to trajectory generation followed by trajectory tracking for this UAV. Finally, the research findings of this paper and possible future work are summarized in section V.

II. Dynamics Model

A. Continuous Time Dynamics

The rigid body model considered in this paper has four control inputs for the six degrees of freedom. These control inputs are three control inputs that generate a torque for the three degrees of freedom of rotational motion, and one thrust along a body-fixed thrust vector as shown in the figure 1. This model is identical to that used in [16, 19]. This

model can be applied to several unmanned vehicles, and the particular case of a quadrotor UAV is considered in section IV for numerical results. *Pose* is the combination of position and orientation of a rigid body and can be represented as:

$$G = \begin{bmatrix} \mathcal{R} & b \\ 0 & 1 \end{bmatrix} \in \mathrm{SE}(3), \tag{1}$$

where $b \in \mathbb{R}^3$ is the rigid body's position vector expressed in an inertial coordinate frame and $\mathcal{R} \in \mathrm{SO}(3)$ is the rigid body's attitude (orientation) expressed as the rotation matrix from inertial frame to body-fixed frame. Without loss of generality, it is assumed that the thrust vector is along the third body-fixed coordinate frame axis. The translational dynamics equation of motion is:

$$m\dot{v} = mge_3 - fr_3, \tag{2}$$

where m is vehicle's mass, g is the gravitational acceleration, $v \in \mathbb{R}^3$ is the translational velocity in inertial frame, $e_3 = [0,\ 0,\ 1]^T$, $fr_3 \in \mathbb{R}^3$ is the control force vector of magnitude f acting on the body, and r_3 is the unit vector along the third axis of the body-fixed coordinate frame, expressed in the inertial frame. Note that r_3 is also the third column of the rotation matrix $\mathcal{R}$, which varies as the rigid body rotates. Equation (2) can be rewritten as:

$$\dot{v} = ge_3 - \frac{1}{m} fr_3. \tag{3}$$

The velocity kinematics for the translational motion expressed in inertial coordinate frame is simply:

$$\dot{b} = v. \tag{4}$$

In addition,

$$\dot{v} = a \qquad \text{and} \qquad \dot{a} = z, \tag{5}$$

where a is acceleration and z is the derivative of acceleration (jerk). Equations (3), (4) and (5) can be considered as the system equations. We are interested in a continuous snap (4^{th} position derivative) because it is desired for torque level control. By defining input $u = fr_3$, state space representation of the system can be rewritten from the above equations in the matrix form, as follows:

$$\dot{x} = Ax + Bu + \begin{bmatrix} 0_{3\times1} \\ ge_3 \\ 0_{3\times1} \\ 0_{3\times1} \end{bmatrix}, \tag{6}$$

where

$$x = \begin{bmatrix} b_{3\times1} \\ v_{3\times1} \\ a_{3\times1} \\ z_{3\times1} \end{bmatrix} \in \mathbb{R}^{12}, \quad u \in \mathbb{R}^3, \quad A = \begin{bmatrix} 0_3 & I_3 & 0_3 & 0_3 \\ 0_3 & 0_3 & 0_3 & 0_3 \\ 0_3 & 0_3 & 0_3 & I_3 \\ 0_3 & 0_3 & 0_3 & 0_3 \end{bmatrix} \in \mathbb{R}^{12\times12}, \quad B = \begin{bmatrix} 0_3 \\ \frac{-1}{m} I_3 \\ 0_3 \\ I_3 \end{bmatrix} \in \mathbb{R}^{12\times3}, \tag{7}$$

subject to

$$x(T_M) = x_{T_M} \ \text{ (given) and } \ T_M : \text{free (unspecified)},$$

where T_M is the unspecified terminal time. A trajectory has to be generated online and onboard the vehicle to go from the given first waypoint at time $t = 0$ (without loss of generality), to the desired final waypoint during the free time duration T_M. In other words, T_M is an additional control parameter. The problem boils down to creating a C^4 trajectory, that is continuous on $\mathbb{R}^{12}$, for the system (6). The solution of this problem can be obtained with a similar treatment to that of section 2.7 of [18]. The novelty of this paper is the discrete time approach to trajectory planning with free final time, and introducing the discrete transversality condition. The discretization facilitates numerical simulation of the system and possible onboard implementation. Hence, the dynamics expressed as the equation (6) will be discretized.

B. Discretization of Dynamics

Considering state space system (6), note that ge_3 term is a constant that can be neglected for discretization as it can be compensated directly by the control thrust vector u. For discretization of the continuous time system $\dot{x} = Ax + Bu$, we have

$$A^d = e^{Ah}, \quad B^d = (\int_0^h e^{A\sigma} d\sigma).B, \tag{8}$$

and $h = t_{i+1} - t_i$. More details of the above discretization can be found in section 4.2.1 of [20]. Note that A^4 is zero, so A^d and B^d can be obtained in exact form. Then discretized system can be obtained as:

$$x_{i+1} = A^d x_i + B^d u_i, \tag{9}$$

where $x_i = [b_i^T,\ v_i^T,\ a_i^T,\ z_i^T]^T \in \mathbb{R}^{12}$ and $u_i \in \mathbb{R}^3$. The trajectory planning problem is explained in the next section.

III. Optimal Trajectory Planning In Discrete Time Setting

The optimal trajectory generation problem consists of constructing a feasible discrete-time trajectory that starts from the initial waypoint x_0 and goes to the final waypoint $x_{N_m} = x_{T_M}$, minimizing a quadratic cost on the states and the control. Here N_M marks the unspecified terminal step. Notice that N_M in discrete setting is equivalent to T_M in continuous setting. Therefore, N_M, the free final step, is the additional control parameter that needs to be selected to optimize the position trajectory. Additionally, the value of final step N_m that gives the optimal trajectory in terms of the quadratic cost needs not be the minimum. The above problem can be solved to find the control parameter N_M and control law u_i through the following steps: First, discrete time Lagrangian is defined as:

$$\mathcal{L}_i^d := \frac{1}{2}(x_i^T Q_i x_i + u_i^T R_i u_i), \tag{10}$$

where $Q_i \geqslant 0$ and $R_i > 0$ are square matrices of appropriate size. Performance index can be defined as:

$$\mathcal{J}^d := \sum_{i=0}^{N_M-1} \mathcal{L}_i^d(x_i, u_i)h + \frac{1}{2}(x_{N_M} - x_{T_M})^T S(x_{N_M} - x_{T_M}), \tag{11}$$

where S is a matrix with appropriate dimensions. By adjusting S, we can tune how hard the constraint enforcing x_{N_M} to x_{T_M} is.

Now to consider system equation (9), the augmented performance index can be written as:

$$\mathcal{J}_a^d := \sum_{i=0}^{N_M-1} \mathcal{L}_i^d(x_i, u_i)h + h\lambda_{i+1}^T\left(A^d x_i + B^d u_i - x_{i+1}\right) + \frac{1}{2}(x_{N_M} - x_{T_M})^T S(x_{N_M} - x_{T_M}), \tag{12}$$

where $\lambda_i \in \mathbb{R}^{12}$ is a vector of co-states, considered as Lagrange multipliers in the following variational development. As the next step, Hamiltonian can be defined as

$$\mathcal{H}_i^d = \lambda_{i+1}^T\left(A^d x_i + B^d u_i\right) + \mathcal{L}_i^d. \tag{13}$$

Therefore $\mathcal{J}_a^d$ in terms of Hamiltonian is

$$\mathcal{J}_a^d = \sum_{i=0}^{N_M-1} (h\mathcal{H}_i^d - h\lambda_{i+1}^T x_{i+1}) + \frac{1}{2}(x_{N_M} - x_{T_M})^T S(x_{N_M} - x_{T_M}). \tag{14}$$

One can set the first variation of $\mathcal{J}_a^d$ to zero, to obtain the necessary conditions for optimality. As a result,

$$d\mathcal{J}_a^d = 0,$$

$$\begin{aligned} \Rightarrow \sum_{i=0}^{N_M-1} h\Big\{\frac{\partial \mathcal{H}_i^d}{\partial x_i}\delta x_i + \frac{\partial \mathcal{H}_i^d}{\partial u_i}\delta u_i + \mathcal{H}_{N_M-1}^d\Big|_{N_M-1} dN_M + \frac{\partial \mathcal{H}_i^d}{\partial \lambda_{i+1}}\delta\lambda_{i+1} - \lambda_{i+1}^T\delta x_{i+1} - \delta\lambda_{i+1}^T x_{i+1}\Big\} \\ + \ (x_{N_M} - x_{T_M})^T S dx_{N_M} = 0, \end{aligned} \tag{15}$$

where dN_M is the difference in N_M, in discrete settings. Note that the term $h\mathcal{H}^d_{N_M-1}\Big|_{N_M-1} dN_M$ is evaluated at final step only. Total derivative of x_{N_M}, can be written as sum of its partial derivative, δx_{N_M}, and $hv_{N_M} dN_M$ (see Fig.2). Note that v_{N_M} is the discrete equivalent of $\dot{x}$, which is in continues setting. Hence, for dx_{N_M}, one can write:

$$dx_{N_M} = \delta x_{N_M} + hv_{N_M} dN_M,$$

and that leads to

$$\delta x_{N_M} = dx_{N_M} - hv_{N_M} dN_M. \tag{16}$$

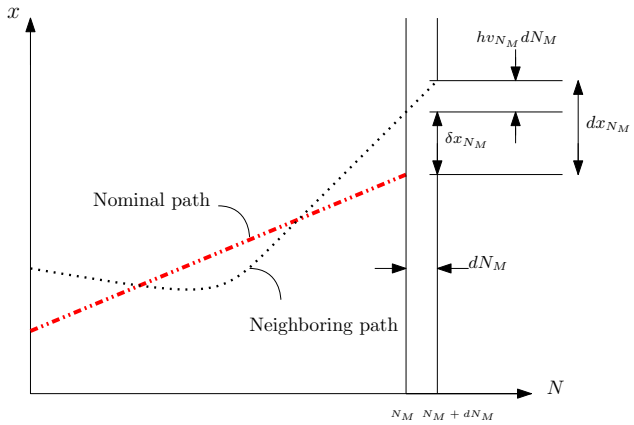


Fig. 2 x**(state) vs** N**(step)**

Figure 2 is the discrete equivalent of Fig 2.7.1 of [18] that depicts the free final time case for continuous time optimal control. Note that dN_M is an integer, because the index set is a subset of the set of integers. Using equation (16), the term $\lambda^T_{i+1}\delta x_{i+1}$ in (15) when $i = N_M - 1$ can be written as:

$$\lambda^T_{N_M}\delta x_{N_M} = \lambda^T_{N_M} dx_{N_M} - \lambda^T_{N_M} hv_{N_M} dN_M. \tag{17}$$

Now, (15) can be rewritten:

$$\sum_{i=0}^{N_M-1} h\left\{\frac{\partial \mathcal{H}^d_i}{\partial x_i}\delta x_i + \frac{\partial \mathcal{H}^d_i}{\partial u_i}\delta u_i + \frac{\partial \mathcal{H}^d_i}{\partial \lambda_{i+1}}\delta\lambda_{i+1} - \lambda^T_{i+1}\delta x_{i+1}\Big|_{i\neq N_M-1} - \delta\lambda^T_{i+1}x_{i+1}\right\}$$
$$+ \left\{(x_{N_M} - x_{T_M})^T S - \lambda^T_{N_M}\right\} dx_{N_M} + h\left\{\mathcal{H}^d_{N_M-1}\Big|_{N_M-1} + \lambda^T_{N_M} v_{N_M}\right\} dN_M = 0. \tag{18}$$

Setting the coefficient of dN_M to zero gives the condition for finding optimum N_M as following:

$$\lambda^T_{N_M} v_{N_M} + \mathcal{H}^d_{N_M-1}\Big|_{N_M-1} = 0. \tag{19}$$

The above equation is "Discrete Transversality Condition", and along with the following optimal trajectory generation scheme, is the main contribution of this work. Also note that equation (19) is a general result, and can be applied to discrete control systems represented by equation (9) and subjected to the cost function (11) with free terminal time.

The necessary conditions for optimality, also found from (18), are as follows:

$$\begin{aligned}
&\frac{\partial \mathcal{H}_i^d}{\partial u_i} = 0 \ \text{ for all } \ i,\\
&\frac{\partial \mathcal{H}_i^d}{\partial x_i} - \lambda_i = 0, \quad \frac{\partial \mathcal{H}_i^d}{\partial \lambda_{i+1}} - x_{i+1} = 0, \quad \text{for } \ i \neq N_M,\\
&S(x_{N_M} - x_{T_M}) = \lambda_{N_M} \ \text{ for } \quad i = N_M.
\end{aligned}$$

The above conditions lead to:

$$u_i = -R_i^{-1}(B^d)^T\lambda_{i+1} \quad \text{(control input)}, \quad \lambda_i = (A^d)^T\lambda_{i+1} + hQ_ix_i \quad \text{(co-states)}, \quad x_{i+1} = A^dx_i + B^du_i \quad \text{(states)}.$$

For ease of implementation, in the following paragraph we show how transversality condition can be written in terms of x_{N_M} and v_{N_M}. Equation (19) can be rewritten in terms of x_{N_M}:

$$\lambda_{N_M}^T v_{N_M} + \lambda_{N_M}^T\left(A^d x_{N_{M-1}} + B^d u_{N_{M-1}}\right) + \frac{1}{2}\{x_{N_{M-1}}^T Q_{N_{M-1}} x_{N_{M-1}} + u_{N_{M-1}}^T R_{N_{M-1}} u_{N_{M-1}}\} = 0. \tag{20}$$

Knowing $u_i = -R_i^{-1}(B^d)^T\lambda_{i+1}$, above equation is rewritten as:

$$\lambda_{N_M}^T(v_{N_M} + x_{N_M}) + \frac{1}{2}\{x_{N_{M-1}}^T Q_{N_{M-1}} x_{N_{M-1}} + \left(-R_{N_{M-1}}^{-1}(B^d)^T\lambda_{N_M}\right)^T R_{N_{M-1}}\left(-R_{N_{M-1}}^{-1}(B^d)^T\lambda_{N_M}\right)\} = 0. \tag{21}$$

Using $u_i = -R_i^{-1}(B^d)^T\lambda_{i+1}$ again, $x_{N_{M-1}}$ can be written in terms of x_{N_M}:

$$x_{N_M} = A^d x_{N_{M-1}} + B^d u_{N_{M-1}} = A^d x_{N_{M-1}} - B^d R_{N_{M-1}}^{-1}(B^d)^T\lambda_{N_M},$$

replacing λ_{N_M} with $S(x_{N_M} - x_{T_M})$, then we have

$$\begin{aligned}
x_{N_M} &= A^d x_{N_{M-1}} - B^d R_{N_{M-1}}^{-1}(B^d)^T S(x_{N_M} - x_{T_M}),\\
x_{N_{M-1}} &= (A^d)^{-1}\{x_{N_M} + B^d R_{N_{M-1}}^{-1}(B^d)^T S(x_{N_M} - x_{T_M})\}.
\end{aligned} \tag{22}$$

Substituting $x_{N_{M-1}}$ from above and λ_{N_M} by $S(x_{N_M} - x_{T_M})$, equation (21) can be simplified as:

$$\begin{aligned}
&(x_{N_M} - x_{T_M})^T S^T(v_{N_M} + x_{N_M}) +\\
&\frac{1}{2}\left\{\{x_{N_M} + B^d R_{N_{M-1}}^{-1}(B^d)^T S(x_{N_M} - x_{T_M})\}^T((A^d)^{-1})^T Q_{N_{M-1}}(A^d)^{-1}\{x_{N_M} + B^d R_{N_{M-1}}^{-1}(B^d)^T S(x_{N_M} - x_{T_M})\}\right\} +\\
&\frac{1}{2}\{(R_{N_{M-1}}^{-1}(B^d)^T S(x_{N_M} - x_{T_M}))^T((B^d)^T S(x_{N_M} - x_{T_M}))\} = 0.
\end{aligned} \tag{23}$$

Equation (23) shows discrete transversality condition in terms of x_{N_M} and v_{N_M} and is used to find the final step N_M. We now can continue to find the control law u_i. Assuming that the optimal control is in the form $u_i = k_ix_i + \eta_i$, the co-state can be expressed as $\lambda_i = h(P_ix_i + \eta_i)$. Therefore, the optimal control input is:

$$\begin{aligned}
u_i &= -R_i^{-1}(B^d)^T\left(P_{i+1}x_{i+1} + \eta_{i+1}\right) = -R_i^{-1}(B^d)^T\left(P_{i+1}\left(A^dx_i + B^du_i\right) + \eta_{i+1}\right),\\
&\Rightarrow [R_i + (B^d)^T P_{i+1}B^d]u_i = -(B^d)^T(P_{i+1}A^dx_i + \eta_{i+1}),
\end{aligned}$$

and that leads to

$$u_i = -[R_i + (B^d)^T P_{i+1}B^d]^{-1}(B^d)^T(P_{i+1}A^dx_i + \eta_{i+1}). \tag{24}$$

By substituting the expressions for $\lambda_i = h(P_ix_i + \eta_i)$ into the co-states equation, one obtains:

$$P_ix_i + \eta_i = (A^d)^T(P_{i+1}x_{i+1} + \eta_{i+1}) + Q_ix_i = (A^d)^T(P_{i+1}(A^dx_i + B^du_i) + \eta_{i+1}) + Q_ix_i, \tag{25}$$

then substituting u_i from equation (24) leads to

$$P_i x_i + \eta_i = (A^d)^T \{P_{i+1}(A^d x_i - B^d[R_i + (B^d)^T P_{i+1} B^d]^{-1}(B^d)^T (P_{i+1} A^d x_i + \eta_{i+1}) + \eta_{i+1}\} + Q_i x_i. \tag{26}$$

The above equation is true for arbitrary x_i, hence

$$P_i = (A^d)^T P_{i+1} A^d + Q_i - (A^d)^T P_{i+1} B^d \left[R_i + (B^d)^T P_{i+1} B^d\right]^{-1} (B^d)^T P_{i+1} A^d, \tag{27}$$

that is true for $i \neq N_M$ and needs to be solved backwards in time. The remainder of equation (25) gives the expression for solving η_i as following:

$$\eta_i = A^{dT} \{P_{i+1}(-B^d[R_i + (B^d)^T P_{i+1} B^d]^{-1}(B^d)^T \eta_{i+1}) + \eta_{i+1}\} = (A^d)^T \{I - P_{i+1} B^d [R_i + (B^d)^T P_{i+1} B^d]^{-1} (B^d)^T\} \eta_{i+1}, \tag{28}$$

that is true for $i \neq N_M$ and also needs to be solved backwards in time. Finally, for the final waypoint we have:

$$P_{N_M} = \frac{1}{h} S, \quad \eta_{N_M} = \frac{-1}{h} S x_{T_M}. \tag{29}$$

After solving P_i and η_i, and having them for all instances, system states can be determined by solving system equation (9) forwards in time. To avoid large transient controls, we use a "forgetting factor", in which Q_i and R_i can be selected as

$$Q_i = \alpha_i \bar{Q} \quad \text{and} \quad R_i = (k - \alpha_i)\bar{R},$$

where $\bar{Q} \geq 0$ and $\bar{R} > 0$ are constant, $\alpha_i = \alpha(t_i)$, $k > 1$ is a constant and $\alpha(t) \in [0, 1]$ is monotonically increasing, with $\alpha(t_0) = 0$ and $\alpha(t_1) = 1$. A choice for $\alpha(t)$ can be non-dimensionalized time:

$$\alpha(t) = \frac{t - t_0}{t_f - t_0}, \tag{30}$$

which satisfies the conditions mentioned above. This helps $\mathcal{L}_i^d$ to go to zero smoothly.
Noting the ge_3 term in equation (6), thrust force can now be calculated at each step by:

$$f_i = m\|a_i - ge_3\|. \tag{31}$$

This generated trajectory can be used in conjunction with nonlinear tracking algorithms to create a complete integrated guidance and control scheme. In following simulations, we show how this is done.

IV. Application to a Quadcoptor UAV tracking

For simulation purposes, we consider a quadrotor UAV model, generate a free-final-time trajectory for it, and then track the trajectory with position and attitude controllers, as following.

A. Six Degrees of Freedom Tracking Control Simulation

The complete control of a quadrotor UAV has two loops: the outer loop position control (for translational motion) and the inner loop attitude control (for rotational motion). The attitude should change such that the desired thrust direction required to follow the position trajectory is achieved. In this work for the inner loop of attitude control, we use the following controller in equation (32) of [21]:

$$\begin{aligned}\tau =& J\left(Q^{\mathrm{T}}\dot{\Omega}_d - \frac{\kappa H(s_K(Q))}{\left(s_K^{\mathrm{T}}(Q)s_K(Q)\right)^{1-1/p}} w(Q,\omega)\right) + (Q^{\mathrm{T}}\Omega_d)^{\times} J\left(Q^{\mathrm{T}}\Omega_d - \kappa z_K(Q)\right) + \kappa J\left(z_K(Q) \times Q^{\mathrm{T}}\Omega_d\right) \\ &+\kappa J(\omega + Q^{\mathrm{T}}\Omega_d) \times z_K(Q) - k_p s_K(Q) - \frac{L_\Omega \Psi(Q,\omega)}{\left(\Psi(Q,\omega)^{\mathrm{T}} L_\Omega \Psi(Q,\omega)\right)^{1-1/p}},\end{aligned} \tag{32}$$

where

$$\Psi(Q,\omega) = \omega + \kappa z_K(Q), \text{and } H(x) = I - \frac{2(1-1/p)}{x^{\mathrm{T}}x} x x^{\mathrm{T}}. \tag{33}$$

Here τ is the control torque, J denotes inertia matrix, Q is the attitude tracking error, Ω is angular velocity, Ω_d is desired angular velocity and ω is the angular velocity tracking error. The cross product operator matrix: $(\cdot)^\times : \mathbb{R}^3 \to \mathfrak{so}(3)$ is given by [22]:

$$x^\times = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}^\times = \begin{bmatrix} 0 & -x_3 & x_2 \\ x_3 & 0 & -x_1 \\ -x_2 & x_1 & 0 \end{bmatrix}.$$

For the outer loop we utilize a position controller that enables us to generate the desired attitude trajectory for the attitude tracking controller. For this purpose, we use the following position controller in equation (18) of [21] for the outer loop:

$$\mathrm{f} = e_3^{\mathrm{T}} \mathcal{R}^{\mathrm{T}} \left(mge_3 + P\tilde{b} + L_v(\mathcal{R}v - v_d) - m\dot{v}_d \right). \tag{34}$$

Here $P, L_v \in \mathbb{R}^{3\times3}$ are positive definite matrices, $b \in \mathbb{R}^3$ is the UAV's inertial position vector, $\tilde{b} = b - b_d$, and $b_d, v_d \in \mathbb{R}^3$ are the desired inertial position and velocity vectors, respectively.

B. Generating the Desired Trajectory and Its Application in Conjunction with Quadcopter Tracking

To obtain numerical results, the system is simulated using sample values of:

$$m = 4.2 \ \ kg, \quad h = 0.01 \ \ s, \quad k = 10, \quad g = 9.81 \ \ ms^{-2}.$$

This trajectory generation scheme can plan a path through multiple waypoints, even waypoints that are introduced on-the-flight. Here in simulations, we only show its application to two waypoints in order to better present the details and the performance of the scheme. The method can be easily repeated between any two adjacent waypoints, in order to connect multiple waypoints to create a smooth path (since it is minimizing snap). The two position waypoints for the simulation here are arbitrarily chosen as:

$$b_{T_1} = \begin{bmatrix} 3 & 2 & 1 \end{bmatrix}^T, b_{T_M} = \begin{bmatrix} 4 & 3 & 3 \end{bmatrix}^T.$$

Matrices $\bar{Q}$ and $\bar{R}$ are tuning parameters and chosen as:

$$\bar{Q} = 5.5 \times 10^{-3} \ \ I_{12} \ \text{ and } \ \bar{R} = 0.0010 \ \ I_3.$$

Simulating the above sample system in Matlab yields the following value for the optimal number of steps:

$$N_M = 572.$$

Considering the step size to be $h = 0.01s$, the above value of N_M corresponds to $T_M = h.N_M = 5.72s$ for the value of final time. In the following, graphical results of this simulation are presented.

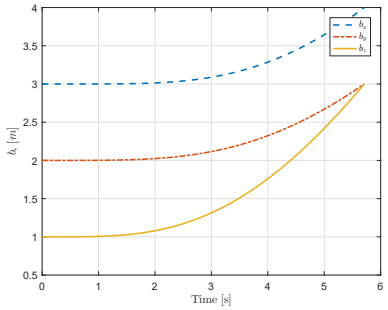


Fig. 3 Position vs time

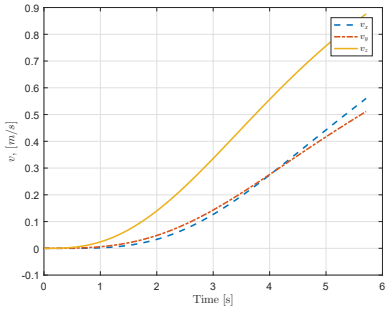


Fig. 4 Velocity vs time

Figure 3 shows position of the vehicle as it starts from initial waypoint and comes to the desired final waypoint in 5.72 seconds. Figure 4 shows associated translational velocity components of the vehicle over time.

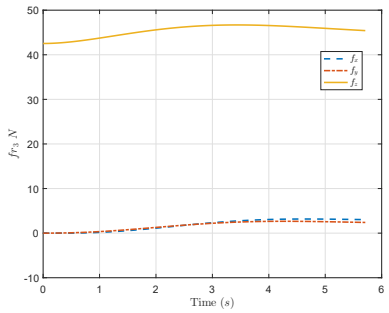


Fig. 5 Thrust components vs time

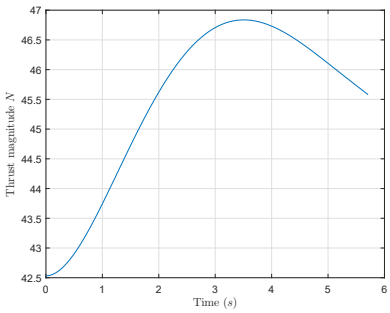
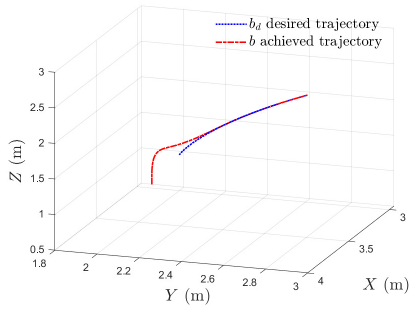


Fig. 6 Thrust force vs time

Figure 5 shows the thrust vector components. Note that the third component is larger than the other two due to gravity. Figure 6 shows reasonable magnitude of thrust needed for the vehicle to realize the generated trajectory in 5.72 seconds. The proposed generated trajectory can be used in conjunction with a nonlinear tracking scheme, such as the one detailed in Section IV.A. This leads to an integrated trajectory generation and tracking scheme with simulation results shown in the following figures, where the vehicle tracks the optimal generated trajectory:



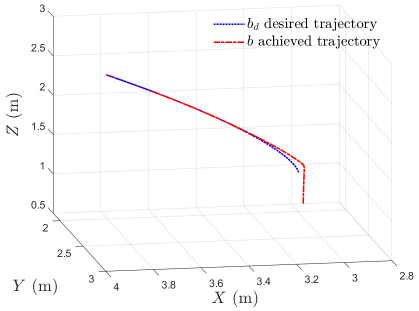
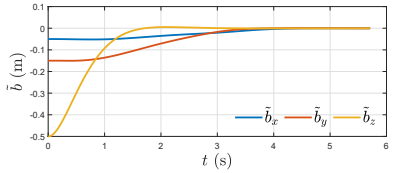


Fig. 7 Tracking the desired trajectory **Fig. 8 Tracking the desired trajectory**

Figures 7 and 8 show the integrated trajectory generation and tracking scheme in the work from different views. The figures show a scenario where a vehicle takes off and converges to an optimal trajectory obtained by the scheme proposed in this paper.



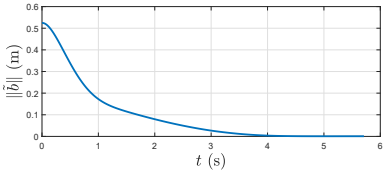


Fig. 9 Components of tracking error **Fig. 10 Norm of the tracking errors**

Figure 9 shows components of position tracking error for the integrated trajectory generation and tracking scheme. Figure 10 presents the norm of position tracking error and how it converges to zero.

V. Conclusions

A novel real-time and onboard-implementable position trajectory planning scheme for an underactuated vehicle, with one thrust control input and three torque control inputs, is proposed. This scheme can generate a trajectory from an initial waypoint to a desired final waypoint in discrete time, while considering the unspecified time duration as an additional control parameter. The unspecified final time is found by using the proposed discrete transversality condition. Numerical simulation results show how the trajectory planning scheme works for a given pair of initial and final position waypoints. This scheme is also numerically simulated in conjunction with a trajectory tracking control scheme for maneuverable unmanned vehicles, like quadrotor UAVs, for integrated onboard trajectory planning and tracking control.

References

[1] Hoy, M., Matveev, A. S., and Savkin, A. V., "Algorithms for collision-free navigation of mobile robots in complex cluttered environments: a survey," *Robotica*, Vol. 34, 2015, pp. 467–497.

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

[3] Sanchez-Lopez, J., Saripalli, S., Campoy, P., Pestana, J., and Fu, C., "Toward visual autonomous ship board landing of a VTOL UAV," *Unmanned Aircraft Systems (ICUAS), 2013 International Conference on*, IEEE, 2013, pp. 779–788.

[4] Stoican, F., and Popescu, D., *Trajectory Generation with Way-Point Constraints for UAV Systems*, Springer International Publishing, 2016.

[5] Gao, F., and Shen, S., "Online quadrotor trajectory generation and autonomous navigation on point clouds," *2016 IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR)*, 2016, pp. 139–146. doi:10.1109/SSRR.2016.7784290.

[6] Cimurs, R., Hwang, J., and Suh, I. H., "Bezier Curve-Based Smoothing for Path Planner with Curvature Constraint," *Robotic Computing (IRC), IEEE International Conference on*, IEEE, 2017, pp. 241–248.

[7] Vitus, M. P., Zhang, W., and Tomlin, C. J., "A hierarchical method for stochastic motion planning in uncertain environments," *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2012, pp. 2263–2268.

[8] Mueller, M. W., Hehn, M., and D'Andrea, R., "A Computationally Efficient Motion Primitive for Quadrocopter Trajectory Generation," *IEEE Transactions on Robotics*, Vol. 31, No. 6, 2015, pp. 1294–1310. doi:10.1109/TRO.2015.2479878.

[9] Li, Y., Eslamiat, H., Wang, N., Zhao, Z., Sanyal, A. K., and Qiu, Q., "Autonomous Waypoints Planning and Trajectory Generation for Multi-Rotor UAVs," *Proceedings of the Workshop on Design Automation for CPS and IoT (DESTION)*, Association for Computing Machinery, New York, NY, USA, 2019, p. 31–40. doi:10.1145/3313151.3313163.

[10] Eslamiat, H., Li, Y., Wang, N., Sanyal, A. K., and Qiu, Q., "Autonomous Waypoint Planning, Optimal Trajectory Generation and Nonlinear Tracking Control for Multi-rotor UAVs," *18th European Control Conference (ECC)*, 2019, pp. 2695–2700. doi:10.23919/ECC.2019.8795855.

[11] Heidari, H., and Saska, M., "Trajectory Planning of Quadrotor Systems for Various Objective Functions," *Robotica*, Vol. 39, No. 1, 2021, p. 137–152. doi:10.1017/S0263574720000247.

[12] Wang, Z., Zhou, X., Xu, C., Chu, J., and Gao, F., "Alternating Minimization Based Trajectory Generation for Quadrotor Aggressive Flight," *IEEE Robotics and Automation Letters*, Vol. 5, No. 3, 2020, pp. 4836–4843. doi:10.1109/LRA.2020.3003871.

[13] Dey, B., and Krishnaprasad, P. S., "Control-theoretic data smoothing," *53rd IEEE Conference on Decision and Control*, 2014, pp. 5064–5070. doi:10.1109/CDC.2014.7040180.

[14] Dey, B., and Krishnaprasad, P. S., "Trajectory smoothing as a linear optimal control problem," *2012 50th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, 2012, pp. 1490–1497. doi:10.1109/Allerton.2012.6483395.

[15] Mellinger, D., Michael, N., and Kumar, V., "Trajectory generation and control for precise aggressive maneuvers with quadrotors," *The International Journal of Robotics Research*, Vol. 31, No. 5, 2012, pp. 664–674.

[16] Viswanathan, S. P., Sanyal, A. K., and Warier, R., "Finite-Time Stable Tracking Control for a Class of Underactuated Aerial Vehicles in SE(3)," *American Control Conference*, Seattle, WA, 2017.

[17] Ma, N., Cao, Y., Wang, X., Wang, Z., and Sun, H., "A Fast path re-planning method for UAV based on improved A* algorithm," *2020 3rd International Conference on Unmanned Systems (ICUS)*, 2020, pp. 462–467. doi:10.1109/ICUS50048.2020.9274912.

[18] Bryson, A., and Ho, Y., *Applied optimal control*, revised ed., Hemisphere Publishing Corporation, Washington, D.C., USA, 1975.

[19] Viswanathan, S. P., Sanyal, A. K., and Izadi, M., "Integrated Guidance and Nonlinear Feedback Control of Underactuated Unmanned Aerial Vehicles in SE(3)," *AIAA Guidance, Navigation, and Control Conference*, Gaylord, TX, 2017. doi: http://dx.doi.org/10.2514/6.2017-1044.

[20] Chen, C.-T., *Linear System Theory and Design*, 2nd ed., Oxford University Press, Inc., New York, NY, USA, 1995.

[21] Viswanathan, S. P., Sanyal, A. K., and Samiei, E., "Integrated Guidance and Feedback Control of Underactuated Robotics System in SE(3)," *Journal of Intelligent & Robotic Systems*, Vol. 89, No. 1, 2018, pp. 251–263.

[22] Sanyal, A., Nordkvist, N., and Chyba, M., "An Almost Global Tracking Control Scheme for Maneuverable Autonomous Vehicles and its Discretization," *IEEE Transactions on Automatic Control*, Vol. 56, No. 2, 2011, pp. 457–462.