

---

# A PRACTICAL RECEDING HORIZON CONTROL FRAMEWORK FOR PATH PLANNING AND CONTROL OF AUTONOMOUS VTOL VEHICLES

---

**C. Liu and W.-H. Chen**

Loughborough University  
Department of Aeronautical and Automotive Engineering,  
Loughborough University, Loughborough LE11 3TU, U.K.

This paper describes an integrated path planning and tracking control framework for autonomous vertical-take-off-and-landing (VTOL) vehicles, particularly quadrotors. The path planning adopts a receding horizon strategy to repeatedly plan a local trajectory that satisfies both the vehicle dynamics and obstacle-free requirement. A tracking controller is then designed to track the planned path. The differential flatness property of the quadrotor is exploited in both path planner and tracking controller designs. The proposed framework is verified by real-time simulations incorporating online optimization.

## 1 INTRODUCTION

Autonomous aerial vehicles, especially those with VTOL ability, such as helicopters, have been found in various applications and have the potentials to execute many complicated tasks. These tasks are very likely to involve agile manoeuvres in a cluttered and obstacle rich environment, which require an autonomous vehicle to be able to replanning a local trajectory that respects not only the obstacle layout but also the vehicle dynamics. To this end, this paper presents an optimization based framework with a receding horizon strategy to guide autonomous VTOL vehicles fly through such an environment so as to achieve the goal position and avoid encountered obstacles.

The implementation of a general task of an autonomous aerial vehicle can be divided into three layers: the high-level mission planning, the mid-level path

planning, and the low-level flight control. The high-level mission planning is beyond the scope of this paper, as it takes in charge of the task allocation to achieve a global goal in a large time scale. The mid-level path planning discussed in this paper is more critical in terms of vehicle's viability, because it has to deal with immediate threats and interact with external environment. Moreover, it is also coupled with the low-level flight control in the sense that the planned trajectory needs to be physically achievable by the vehicle and the replanning needs to be quick enough to cope with external disturbances. As only the information in a local region needs to be taken into account, it is possible to formulate the path planning into a finite-time optimal control problem subject to vehicle dynamics and other constraints, which then can be solved online repeatedly. This strategy is known as receding horizon control (RHC) or model predictive control (MPC) [1].

A quadrotor is considered in this paper to represent the general VTOL dynamics. Due to the dynamic feature of this type of aerial vehicles, the flight control can be further decomposed into an outer-loop for slow translational movement and an inner-loop for fast attitude movement. First, the attitude dynamics of the quadrotor is feedback linearized, so that a linear tracking controller can be designed to track a desired attitude. Next, the simplified dynamics are integrated into the path planner to guarantee the generated trajectory is dynamically feasible for the quadrotor to track. During this process, the differential flatness property is used, which means that all the system states and control input can be expressed in terms of the system output and its derivatives, i. e., in the output space [2]. Path planning using the differential flatness was reported in [3,4]. This paper follows a similar idea to facilitate the application but with a focus on the integration of the path planning and tracking control.

Since the local path planning for quadrotors is formulated into an optimization problem and performed in the output space, the quadrotor path (output trajectory) in the receding horizon can be efficiently described and parametrized by polynomials curves. Moreover, a potential field method is integrated into the optimization process to achieve obstacle avoidance function. Although path planning is performed in the output space, by exploiting the differential flatness property again, the optimized path can be used to calculate the desired attitude and the corresponding trim control input for the tracking controller. Thus, a linear parameter varying (LPV) model can be established based on the error between the desired attitude and the measured one, as well as the trim control input. This model is then governed by an LPV controller designed by using an MPC technique.

Receding horizon control based path planning has been studied by many researchers and has shown promising capabilities in both theory and practice [5–7]. One feature of the proposed RHC framework in this paper is to allow the inclusion of a nonlinear vehicle model into the online optimization without introducing extra computational load. As polynomial curves are adopted to represent

the interested output trajectory directly, the integration of vehicle’s differential equations can be eliminated from the evaluation of the cost function, so that the computational load can be substantially reduced.

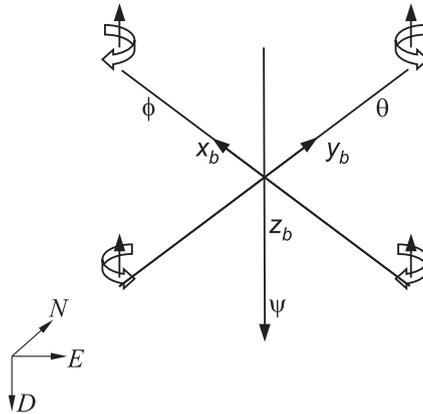
## 2 VEHICLE MODEL

### 2.1 Dynamical Model

The dynamic model of a quadrotor has been studied in many research works [8–10]. Essentially, a quadrotor can be modeled as a rigid-body driven by external forces and torques. The derivation of the nonlinear dynamics is first performed in the body-fixed coordinates  $\mathcal{B}$  using Euler–Newton equations of motion, and then transformed into the  $NED$  inertial frame  $\mathcal{I}$  to facilitate the path planning. Let  $[e_1 e_2 e_3]^T$  denote the unit vector along the corresponding inertial axes, and  $[x_b y_b z_b]^T$  denote unit vector along the respective body axes as shown in Fig. 1.

The position vector is denoted by  $\xi = [x y z]^T$ , which represents the coordinates of the vehicle’s center of gravity relative to the inertial coordinates. The orientation of the vehicle with respect to the inertial coordinate can be parametrised by an Euler angle vector  $\eta = [\phi \theta \psi]^T$  in the aeronautical  $Z$ – $Y$ – $X$  convention, where the transformation matrix  $\mathcal{R}$  is defined as

$$\mathcal{R} = \begin{bmatrix} c\theta c\psi & s\phi s\theta c\psi - c\phi s\psi & c\phi s\theta c\psi + s\phi s\psi \\ c\theta s\psi & s\phi s\theta s\psi + c\phi c\psi & c\phi s\theta s\psi - s\phi c\psi \\ -s\theta & s\phi c\theta & c\phi c\theta \end{bmatrix}.$$



**Figure 1** Quadrotor configuration

In addition,  $\Omega = [p \ q \ r]^T$  represents the angular rate of the vehicle projected in the body-fixed frame. The vehicle dynamics can be obtained by applying the Euler–Newton principle [11]:

$$m\ddot{\xi} = -\mathcal{R}F + mge_3; \quad (1)$$

$$J\dot{\Omega} = -\Omega \times J\Omega + \tau \quad (2)$$

where  $\mathcal{R}$  is the rotation matrix;  $m$  is the vehicle mass;  $g$  denotes the gravitational acceleration;  $J$  describes the inertia tensor matrix;  $F$  is the external force vector; and  $\tau$  is the external torque vector, both expressed in the body-fixed frame.

The external force and torques exerted on a quadrotor are primarily generated by its four rotors. As a VTOL vehicle, it can be observed that the resultant force  $u$  of the four rotors is perpendicular to the vehicle along the  $Z$  direction, such that the total force can be written as  $F = [0 \ 0 \ -u]^T$  in the body-fixed frame. The torque vector  $\tau = [\tau_\phi \ \tau_\theta \ \tau_\psi]^T$  is produced by altering rotor thrusts as a pair in opposite directions. The expression of the forces and torques depends on the rotorcraft configuration. For a quadrotor, a common simplified relationship is adopted by only considering the four rotor thrusts, so that the control inputs ( $u$ ,  $\tau_\phi$ ,  $\tau_\theta$ , and  $\tau_\psi$ ) are altered by the rotor speed ( $w_1$ ,  $w_2$ ,  $w_3$ , and  $w_4$ ) [10]:

$$\begin{bmatrix} u \\ \tau_\phi \\ \tau_\theta \\ \tau_\psi \end{bmatrix} = \begin{bmatrix} \rho & \rho & \rho & \rho \\ 0 & -\rho l & 0 & \rho l \\ -\rho l & 0 & \rho l & 0 \\ \kappa & -\kappa & \kappa & -\kappa \end{bmatrix} \begin{bmatrix} w_1^2 \\ w_2^2 \\ w_3^2 \\ w_4^2 \end{bmatrix}$$

where  $\rho$  and  $\kappa$  are the positive constants characterizing the rotor aerodynamics, and  $l$  denotes the distance from rotors to the center of gravity.

To facilitate the following path planning and control design, the attitude dynamics of the quadrotor are further explored. Based on the Euler angle parametrization, the angular kinematic relationship is described by the following equation:

$$\dot{\eta} = \Phi(\eta)\Omega \quad (3)$$

where

$$\Phi(\eta) = \begin{bmatrix} 1 & \sin \phi \tan \theta & \cos \phi \tan \phi \\ 0 & \cos \phi & -\sin \phi \\ 0 & \sin \phi \sec \theta & \cos \phi \sec \theta \end{bmatrix}.$$

This relationship implies  $\Omega = \Psi(\eta)\dot{\eta}$  where  $\Psi(\eta) = \Phi^{-1}(\eta)$ . Next, differentiating Eq. (3) and invoking the angular dynamics (2), the following relationship can be obtained,

$$\ddot{\eta} = \dot{\Phi}\Omega + \Phi\dot{\Omega} = \dot{\Phi}\Psi\dot{\eta} - \Phi J^{-1}\Omega \times J\Omega + \Phi J^{-1}\tau. \quad (4)$$

Further manipulating Eq. (4) by multiplying  $\Psi(\eta)^T J \Psi(\eta)$  on both sides gives:

$$M(\eta)\ddot{\eta} + C(\dot{\eta}, \eta)\dot{\eta} = \Psi(\eta)^T \tau \quad (5)$$

where

$$\begin{aligned} M(\eta) &= \Psi(\eta)^T J \Psi(\eta); \\ C(\dot{\eta}, \eta) &= -\Psi(\eta)^T J \dot{\Psi}(\eta) + \Psi(\eta)^T \Psi(\eta) \times \dot{\eta} J \Psi(\eta). \end{aligned}$$

By observing Eq. (5), it is possible to define a pseudocontrol input  $\tilde{\tau} = [\tilde{\tau}_\phi \ \tilde{\tau}_\theta \ \tilde{\tau}_\psi]^T$  such that the actual torque is expressed as

$$\tau = J \Psi(\eta) \tilde{\tau} + \Phi^T C(\eta, \dot{\eta}) \dot{\eta}. \quad (6)$$

Next, by substituting Eq. (6) into Eq. (1), the quadrotor dynamics can be expressed by:

$$\begin{aligned} \ddot{x} &= -\frac{1}{m} u (\cos(\phi) \sin(\theta) \cos(\psi) + \sin(\phi) \sin(\psi)); \\ \ddot{y} &= -\frac{1}{m} u (\cos(\phi) \sin(\theta) \sin(\psi) - \sin(\phi) \cos(\psi)); \\ \ddot{z} &= -\frac{1}{m} u \cos(\phi) \cos(\theta) + g; \\ \ddot{\phi} &= \tilde{\tau}_\phi; \quad \ddot{\theta} = \tilde{\tau}_\theta; \quad \ddot{\psi} = \tilde{\tau}_\psi. \end{aligned} \quad (7)$$

In this model, the system state is  $\mathbf{x} = [x \ y \ z \ \dot{x} \ \dot{y} \ \dot{z} \ \phi \ \theta \ \psi \ \dot{\phi} \ \dot{\theta} \ \dot{\psi}]^T$ , the control input is defined as  $\mathbf{u} = [u \ \tilde{\tau}_\phi \ \tilde{\tau}_\theta \ \tilde{\tau}_\psi]^T$ , and the system output interested in path planning is  $\mathbf{z} = [x \ y \ z \ \psi]^T$ .

## 2.2 Differential Flatness Model

Path planning using receding horizon techniques needs a model of the vehicle to predict its future behavior. However, directly using the full dynamic equations (7) is computationally intensive even if a finite time optimal control problem is considered in the receding horizon framework [12, 13]. However, by exploring the differential flatness property of the quadrotor dynamics, it is possible to take into account the vehicle dynamics in the path planning without introducing extra complications.

By definition, differential flatness is a property that for a nonlinear system  $\dot{\mathbf{x}} = f(\mathbf{x}, \mathbf{u})$ , with the output  $\mathbf{z} = y(\mathbf{x})$ , all of its states and inputs can be determined in terms of the output vector  $\mathbf{z}$  and a finite number of its time derivatives [2], such that

$$\mathbf{x} = g(\mathbf{z}, \dot{\mathbf{z}}, \ddot{\mathbf{z}}, \dots, \mathbf{z}^{[n]}); \quad \mathbf{u} = h(\mathbf{z}, \dot{\mathbf{z}}, \ddot{\mathbf{z}}, \dots, \mathbf{z}^{[n+1]}). \quad (8)$$

Differential flatness provides two important features for the path planning and tracking control for quadrotors. First, the path planning can be performed in

the output space by manipulating vehicle's position and heading angle directly. Second, after the optimal trajectory is found, the corresponding vehicle attitude and control input can be calculated instantaneously.

For a quadrotor with dynamics described in Eq. (7), the investigation on its differential flatness property can start from the following relationship:

$$\left. \begin{aligned} \phi &= \sin^{-1} \left( \frac{m}{u} (-\ddot{x} \sin \psi + \ddot{y} \cos \psi) \right); \\ \theta &= \tan^{-1} \left( -\frac{\ddot{x} \cos \psi + \ddot{y} \sin \psi}{g - \ddot{z}} \right); \\ \psi &= \psi. \end{aligned} \right\} \quad (9)$$

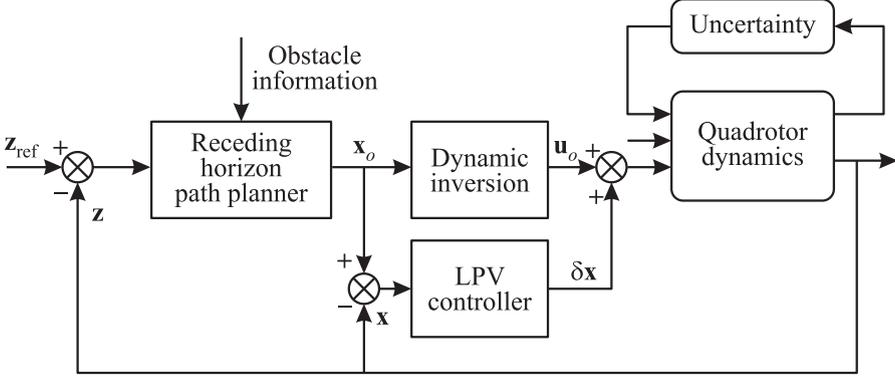
On the other hand, the control inputs can be represented by:

$$\left. \begin{aligned} u &= \sqrt{\ddot{x}^2 + \ddot{y}^2 + (\ddot{z} - g)^2}; \\ \tilde{\tau}_\phi &= \ddot{\phi}; \quad \tilde{\tau}_\theta = \ddot{\theta}; \quad \tilde{\tau}_\psi = \ddot{\psi}. \end{aligned} \right\} \quad (10)$$

The higher derivatives of  $\phi$ ,  $\theta$ , and  $\psi$  can also be derived from Eqs. (9) and (10) by continuous differentiation and substitution. At this stage, it is useful to use a software toolbox that supports the symbolical operation (for example, the Matlab Symbolic toolbox) to facilitate the derivation. The state and control of the quadrotor can eventually be expressed in the output space in the format of Eqs. (8).

### 3 PATH PLANNING AND TRACKING CONTROL

The framework suggested in this paper solves the path planning and tracking control for autonomous vehicles in a hierarchical structure. A path planner first generates a reference trajectory that is obstacle-free according to the environment information and dynamically feasible for the vehicle to track. Then, the planned reference trajectory is propagated to the tracking controller which governs the vehicle dynamics to track the given trajectory. This framework is shown in Fig. 2. More specifically, the path planner works in a receding horizon fashion to incorporate the newly detected environment information and to produce a feasible trajectory using online optimization. The generated state trajectory  $\mathbf{x}_o$  from path planner is then fed into a dynamics inversion block that uses differential flatness property to calculate the corresponding trim control  $\mathbf{u}_o$ . To compensate uncertainties and reject external disturbances in the vehicle dynamics, a linear parameter varying controller is adopted to generate the control compensation  $\delta \mathbf{u}$  based on the difference between an actual vehicle state and the reference one.



**Figure 2** Path planning and tracking control configuration

### 3.1 Receding Horizon Path Planning

The generation of the optimal trajectory in a finite time horizon  $t \in [t_0, t_0 + T]$  that satisfies a number of constraints, such as vehicle dynamics, control saturation, and obstacles en-route, can be formulated as an optimization problem, such that

$$\mathbf{x}_o, \mathbf{u}_o = \arg \min_{\mathbf{x}, \mathbf{u}} J(\mathbf{x}(t), \mathbf{u}(t)) \quad (11)$$

subject to

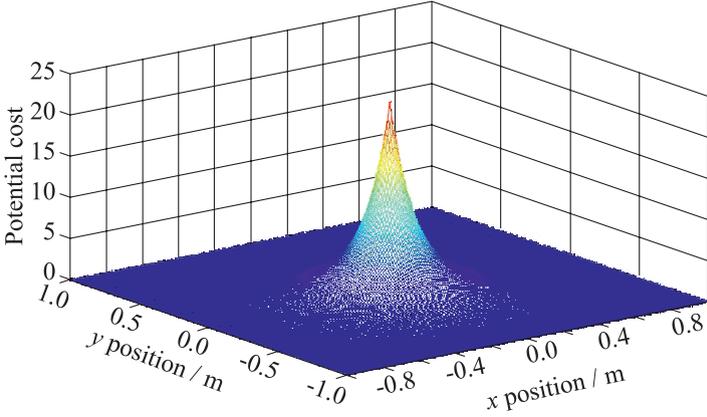
$$\begin{aligned} \dot{\mathbf{x}}(t) &= f(\mathbf{x}(t), \mathbf{u}(t)); \\ \mathbf{z}(t) &= y(\mathbf{x}(t)); \quad \mathbf{x}(t) \in \mathbb{X}; \\ \mathbf{u}(t) &\in \mathbb{U}; \quad \mathbf{x}(t_0 + T) \in \mathbb{X}_f \end{aligned} \quad (12)$$

where  $\mathbf{x}_o$  is the optimized state trajectory;  $\mathbf{u}_o$  is the corresponding optimal input;  $J(k)$  is the cost function to be minimized, Eq. (12) is the vehicle dynamics,  $\mathbb{X}$  and  $\mathbb{U}$  are the state and control input constraint, respectively, and  $\mathbb{X}_f$  is the terminal constraints usually used to guarantee the stability property [1].

The path planning considered in this paper belongs to the local path planning, which means that it needs to follow a global reference and replans a local trajectory if necessary for a short term future. In this case, the cost function  $J$  can be stated as:

$$J(t_0) = \int_{t_0}^{t_0+T} (\mathbf{z}_{\text{ref}}(t) - \mathbf{z}(t))^T Q (\mathbf{z}_{\text{ref}}(t) - \mathbf{z}(t)) + J_{\text{obs}}(t) dt \quad (13)$$

where  $\mathbf{z}_{\text{ref}}$  is the global path reference;  $\mathbf{z}$  is the vehicle output;  $Q$  is the positive definite weighting matrix; and  $J_{\text{obs}}(t)$  is the obstacle cost penalty. The local



**Figure 3** Potential field about a point at  $(0, 0)$ .

path planning formulation can be extended to a global one by incorporating appropriate cost-to-go penalties in the cost functions [14, 15].

The obstacle penalty  $J_{\text{obs}}$  may consist of several contributors such as  $J_{\text{obs}} = \sum_{i=1}^n J_{\text{obs}}^i$  where  $n = 1, 2, \dots$  is the number of obstacles being considered. For each obstacle, the penalty cost can be provided by a potential function like a Yukawa function:

$$J_{\text{obs}}^i = \beta \frac{e^{\alpha d_i}}{d_i + \epsilon}, \quad i = 1, \dots, n, \quad (14)$$

where  $\beta$  is the scaling factor;  $\alpha$  is the decay rate of the potential field;  $d_i$  is the distance between the helicopter and the nearest point on the  $i$ th obstacle; and  $\epsilon$  is the small positive scalar to prevent singularity.

An example of two-dimensional (2D) potential field around a point obstacle is shown in Fig. 3, where it can be seen that the penalty cost approaches infinity as the distance to the obstacle gets close to zero. Acceptable safe clearance distance can be defined using the potential field design parameters  $\alpha$  and  $\beta$ . For a detected obstacle with location  $(x_{\text{obs}}, y_{\text{obs}}, z_{\text{obs}})$  and a safety distance  $r_{\text{obs}}$ , the distance  $d_i$  can be calculated as

$$d_i = \sqrt{(x - x_{\text{obs}})^2 + (y - y_{\text{obs}})^2 + (z - z_{\text{obs}})^2} - r_{\text{obs}}.$$

By incorporating the potential term of Eq. (14), the overall cost function (13) can be seen as a trade-off performance index for tracking a predefined global reference and diverging from obstacles.

The optimization problem stated in Eq. (11) with the cost function defined in Eq. (13) needs to be solved in each sampling instant. The computational

load involved in this procedure is very heavy, as it is a nonconvex optimization problem and involves integration of the system dynamics (12) of 12 orders.

### 3.2 Trajectory Optimization Using Polynomials

As mentioned before, solving the optimization problem stated in Eq. (11) is computationally intensive and may result in a low sampling rate that cannot response fast enough to newly detected obstacles. To overcome this problem, this optimization problem is converted into the output space by using the differential flatness property and polynomial curves, such that:

$$\min_{\mathbf{x}, \mathbf{u}} J(\mathbf{x}, \mathbf{u}) \Leftrightarrow \min_{\mathbf{x}(\mathbf{z}), \mathbf{u}(\mathbf{z})} J(\mathbf{x}(\mathbf{z}), \mathbf{u}(\mathbf{z})) \Leftrightarrow \min_{\mathbf{z}} J(\mathbf{z}).$$

Thus, the original problem (11) is equivalent to the following optimizations problem:

$$\min_{\mathbf{z}} J(\mathbf{z}) \tag{15}$$

subject to

$$g(\mathbf{z}(t)) \in \mathbb{X}; \quad h(\mathbf{z}(t)) \in \mathbb{U}; \quad g(\mathbf{z}(t_0 + T)) \in \mathbb{X}_f.$$

It can be noted that in optimization problem (15), the integration process of the nonlinear system is eliminated.

To deal with the infinite dimension of the problem (15), a suitable output space parametrisation is required to transform it to a computationally tractable nonlinear programming (NLP). A common approach is to use polynomial function to approximate the output trajectories as a linear combination of constant coefficients and basis function, so that for each output

$$z_j(t) = \sum_{i=0}^n B_i(t) P_{j,i}$$

where  $j = 1, 2, 3, 4$  denotes the  $j$ th element in  $\mathbf{z}$ ;  $B_i(t)$  is the  $i$ th order basis function;  $P_{j,i}$  is the coefficient the of  $i$ th order basis function; and  $n$  is the order of the polynomials.

In this paper, the Bezier polynomials are chosen to approximate the quadrotor output trajectory. The basis function of Bezier polynomials is defined as

$$B_{i,n}(\tau) = \frac{n!}{i!(n-i)!} \tau^i (1-\tau)^{n-i}, \quad \tau \in [0, 1],$$

where  $\tau$  is the polynomial curve parameter. Since the optimization in the receding horizon framework covers a fixed time period  $T$ , it is convenient to link the time index to the curve parameter such that  $t = \tau T$ . The fixed time horizon

allows the rapid calculation of the time derivatives of the polynomial trajectory which consequently facilitates the expression of the differential flatness system. This can be seen from the following equation:

$$\frac{dB_i(\tau)}{dt} = \frac{dB_i(\tau)}{d\tau} \frac{d\tau}{dt} = \frac{1}{T} \frac{dB_i(\tau)}{d\tau}.$$

Therefore, the derivatives of the output trajectory can be easily represented:

$$z_j^{[r]}(\tau) = \sum_{i=0}^n \frac{1}{T^r} B_{i,n}^{[r]}(\tau) P_{j,i}$$

where  $B_{i,n}^{[r]}(\tau)$  is the  $r$ th derivative of the basis function that can be calculated offline by using the de Casteljau algorithm [16]. Moreover, the initial state of vehicle can be enforced in the NLP by assigning the coefficients in polynomial curves. Suppose  $\mathbf{z}(0)$ ,  $\dot{\mathbf{z}}(0)$ , and  $\ddot{\mathbf{z}}(0)$  can be provided by the onboard sensor suit for each online optimization, the corresponding coefficients  $P_{j,0}$ ,  $P_{j,1}$ , and  $P_{j,2}$ ,  $j = 1, 2, 3, 4$ , can be determined by using the following relationships:

$$\left. \begin{aligned} P_{j,0} &= z_j(0); \\ P_{j,1} &= \frac{\dot{z}_j(0)}{n} T + P_{j,0}; \\ P_{j,2} &= \frac{\ddot{z}_j(0)}{(n-1)n} T^2 - P_{j,0} + 2P_{j,1}. \end{aligned} \right\} \quad (16)$$

These relationships reduce the number of coefficients that need to be optimized, hence, reduce the dimension of the final optimization problem.

The optimization problem (15) is finally converted into an NLP by using polynomial parameterization:

$$\min_{P_{j,i}} J(P_{j,i}) \quad (17)$$

where  $j = 1, 2, 3, 4$  corresponds to the four interested outputs in the path planning; and  $i = 0, 1, \dots, n$  is decided by the order of the polynomials. Since the first three coefficients in each output trajectory can be predetermined by Eqs. (16), the actual number of variables to be optimized in the online optimization is  $j(n+1-3)$ . In addition, the cost function is defined as

$$J = \sum_{\tau_k=0}^{\tau_N=1} \|\mathbf{z}_{\text{ref}}(\tau_k) - \mathbf{z}(\tau_k)\|^2 + J_{\text{obs}}(\tau_k)$$

where  $k$  denotes the collocation point and  $N$  is the total number of collocation points, such that  $\tau_i = 0, \tau/N, \dots, 1$ . It can be noted that the resolution of the optimization problem (17) depends on the order of the polynomials. Therefore, there is a trade-off between the accuracy of the solution and the computational efficiency.

### 3.3 Tracking Control Design

After the path planning is solved in the receding horizon framework, a tracking controller is required to control the vehicle to track such a trajectory. Given the optimized trajectory  $\mathbf{z}_o$  from the path planner, the corresponding optimized state  $\mathbf{x}_o$ , and control inputs  $\mathbf{u}_o$  can be calculated based on the differential flatness property in Eqs. (8). However, due to the mismatching between the mathematical model and the real quadrotor dynamics, the noises, and disturbances in the process, this kind of optimal control may result in a significantly degraded performance. Therefore, it is essential to design a corrective input term  $\delta\mathbf{u}$  based on the gap between the optimal state  $\mathbf{x}_o$  and the actual system state  $\mathbf{x}$  [17].

The feedback tracking controller is designed based on perturbation models around the optimal state  $\mathbf{x}_o$  and control  $\mathbf{u}_o$ . The quadrotor model can be linearized around the nominal reference and input as:

$$\dot{\mathbf{x}} = f(\mathbf{x}, \mathbf{u}) \approx f(\mathbf{x}_o, \mathbf{u}_o) + \left. \frac{\partial f}{\partial \mathbf{x}} \right|_{\mathbf{x}_o, \mathbf{u}_o} (\mathbf{x} - \mathbf{x}_o) + \left. \frac{\partial f}{\partial \mathbf{u}} \right|_{\mathbf{x}_o, \mathbf{u}_o} (\mathbf{u} - \mathbf{u}_o). \quad (18)$$

By defining the error state  $\delta\mathbf{x} = \mathbf{x} - \mathbf{x}_o$  and control compensation  $\delta\mathbf{u} = \mathbf{u} - \mathbf{u}_o$ , the system (18) can be stated as a linear parameter varying system:

$$\delta\dot{\mathbf{x}} = \left. \frac{\partial f}{\partial \mathbf{x}} \right|_{\mathbf{x}_o, \mathbf{u}_o} \delta\mathbf{x} + \left. \frac{\partial f}{\partial \mathbf{u}} \right|_{\mathbf{x}_o, \mathbf{u}_o} \delta\mathbf{u} = A_o(\mathbf{x}_o, \mathbf{u}_o)\delta\mathbf{x} + B_o(\mathbf{x}_o, \mathbf{u}_o)\delta\mathbf{u}. \quad (19)$$

There are many synthesis techniques for designing LPV controllers once an LPV model is known, such as gain scheduling control [18],  $H_\infty$  control [17], and model predictive control [19]. This paper uses a classic MPC technique introduced in [20] to achieve the regulation of the LPV system.

For system (19), a linear MPC is employed to regulate the state  $\delta\mathbf{x}$  to zero, where the performance index is specified by a quadratic cost function to be minimized:

$$J(k) = \frac{1}{2} \sum_{i=1}^{H_p} \|\delta\mathbf{x}_{k+i}\|_Q^2 + \frac{1}{2} \sum_{i=0}^{H_c-1} \|\delta\mathbf{u}_{k+i}\|_R^2 \quad (20)$$

where  $k$  indicates the time step at which the state is updated; and  $\delta\mathbf{x}_{k+i}$ ,  $i = 1, \dots, H_p$ , is the  $i$ th step ahead prediction of the error state with  $H_p$  denoting the prediction horizon. The predictions of tracking errors are the functions of the future control  $\delta\mathbf{u}_{k+i}$ ,  $i = 1, \dots, H_c - 1$ , where  $H_c$  is the control horizon, beyond which the control keeps the same value.

A reformulation can be performed by considering the system state in a matrix form  $\bar{\mathbf{X}} = [\delta\mathbf{x}_1^T, \delta\mathbf{x}_2^T, \dots, \delta\mathbf{x}_{H_p}^T]^T$ . For each element  $\delta\mathbf{x}_i$ , the evolution of system (19) can be represented by

$$\delta\mathbf{x}_i = \hat{A}_o^i \delta\mathbf{x}_0 + \sum_{j=0}^{i-1} \hat{A}_o^j \hat{B}_o \delta\mathbf{u}_{i-1-j}, \quad i = 1, 2, \dots, H_p$$

where  $\hat{A}_o$  and  $\hat{B}_o$  denote their discrete counterparts. Thus, a matrix expression of the evolution of all the states of the system can be derived as

$$\bar{X} = \bar{A}\delta\mathbf{x}_0 + \bar{B}\bar{U} \quad (21)$$

where

$$\bar{U} = [\delta\mathbf{u}_0^T, \delta\mathbf{u}_1^T, \dots, \delta\mathbf{u}_{N_c-1}^T]^T$$

with the corresponding dimension, and

$$\bar{A} = \begin{bmatrix} \hat{A} \\ \hat{A}_o^2 \\ \vdots \\ \hat{A}_o^{N_c-1} \\ \vdots \\ \hat{A}_o^{N_p} \end{bmatrix}; \quad \bar{B} = \begin{bmatrix} \hat{B}_o & 0 & \cdots & 0 \\ \hat{A}_o\hat{B}_o & \hat{B}_o & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots \\ \hat{A}_o^{H_c-1}\hat{B}_o & \hat{A}_o^{H_c-2}\hat{B}_o & \cdots & \hat{B}_o \\ \vdots & \vdots & \vdots & \vdots \\ \hat{A}_o^{H_p-1}\hat{B}_o & \hat{A}_o^{H_p-2}\hat{B}_o & \cdots & \hat{A}_o^{H_p-H_c}\hat{B}_o \end{bmatrix}.$$

Therefore, by inserting Eq. (21) into Eq. (20), the cost function can be written as the following formulation:

$$J^*(\delta\mathbf{x}_0) = \frac{1}{2} \delta\mathbf{x}_0^T Y \delta\mathbf{x}_0 + \min_{\bar{U}} \left\{ \frac{1}{2} \bar{U}^T H \bar{U} + \delta\mathbf{x}_0^T F \bar{U} \right\}$$

where  $J^*$  is the optimal cost as a function of initial state  $\delta\mathbf{x}_0$ ;  $Y = \bar{A}^T \bar{Q} \bar{A}$ ;  $H = \bar{B}^T \bar{Q} \bar{B} + \bar{R}$ ; and  $F = \bar{A}^T \bar{Q} \bar{A}$ , in which  $\bar{Q} = \text{diag}\{Q, \dots, Q\}$  and  $\bar{R} = \text{diag}\{R, \dots, R\}$  in the corresponding dimensions. Using the optimality condition  $\partial J / \partial \bar{U} = 0$ , the control input can be calculated as

$$\bar{U} = \frac{1}{2} H^{-1} F \delta\mathbf{x}_0$$

where the first element  $\delta\mathbf{u}_0$  is applied to formulate the composite control input  $\mathbf{u} = \mathbf{u}_o + \delta\mathbf{u}_o$ .

## 4 SIMULATION

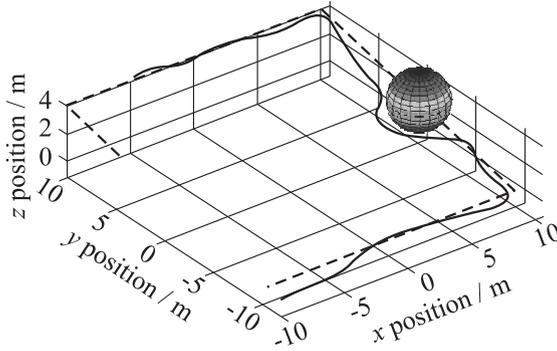
Numerical simulations are carried out to verify the suggested hierarchical path planning and tracking framework. The implementation of the proposed receding horizon framework is achieved by using two computers running the Simulink and Matlab environment. One computer uses Simulink to execute the tracking controller and a detailed quadrotor dynamic model, whereas another computer is used as a path planner to solve the formulated nonlinear programming by

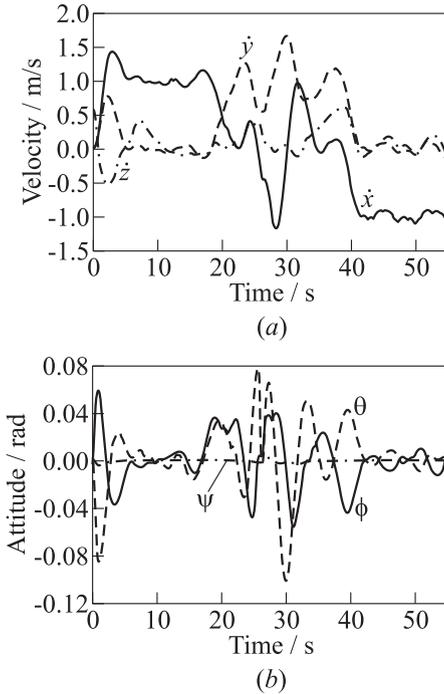
**Table 1** Parameters for simulation

| Notation | Description                  | Value |
|----------|------------------------------|-------|
| $n$      | Order of polynomials         | 6     |
| $T$      | Prediction horizon           | 10 s  |
| $N$      | Collocation points           | 20    |
| $f$      | Sampling rate of RHC planner | 5 Hz  |
| $A$      | Potential field scaler       | 60    |
| $\alpha$ | Potential field decay rate   | -5    |

using the Matlab `fmincon` function. The communication between two computers relies on the local area network (LAN) using the UDP/IP (User Datagram Protocol/Internet Protocol). Moreover, both simulations are synchronized and performed in real time to include the computational delay arisen from online optimization. The path planner parameters used in the simulation are summarized in Table 1. Regarding the MPC tracking controller, the horizon is chosen as  $H_p = H_c = 10$  and the weight matrices are designed with appropriate dimensions.

A primary simulation result is presented here which is to track a multisection trajectory while avoiding a sphere obstacle. The tracking result is given in Fig. 4. The planner can predict 10-second future path in an updating rate of 5 Hz on a 2.66-gigahertz personal computer, but the calculation can be significantly accelerated if executed in a compiled C code. It can be seen that the quadrotor is able to track the predetermined trajectory with a good accuracy and avoid the obstacle encountered. The corresponding velocity and attitude profiles are given in Fig. 5, which show the smooth transition during the avoidance manoeuvre and cornering.

**Figure 4** Trajectory tracking result



**Figure 5** Velocity and attitude profiles

## 5 SUMMARY AND FUTURE WORK

This paper discusses a hierarchical path planning and tracking control framework for an autonomous quadrotor using a practical receding horizon control strategy. The differential flatness property of the quadrotor dynamics is first exploited in the path planning to transfer the path optimization from input space to the output space and to combine with polynomial parametrization to enforce the vehicle dynamic constraints. Furthermore, the differential flatness is utilized to generate the optimal state and control based on the optimized trajectory. These optimal state and control are incorporated with an LPV controller to eventually achieve the trajectory tracking of the quadrotor.

The suggested path planning and control framework is evaluated via numerical simulations, which show promising results. The hardware-in-the-loop simulation and flight experiment using an AscTec Hummingbird quadrotor are also under consideration to further explore the capability of the proposed planning and control framework.

## REFERENCES

1. Mayne, D. Q., J. B. Rawlings, C. V. Rao, and P. O. M. Scokaert. 2000. Constrained model predictive control: Stability and optimality. *Automatica* 36(6):789–814.
2. Fliess, M., J. Lévine, P. Martin, and P. Rouchon. 1995. Flatness and defect of non-linear systems: Introductory theory and examples. *Int. J. Control* 61(6):1327–61.
3. Berry, A., J. Howitt, I. Postlethwaite, and D. Gu. 2009. Situation aware trajectory tracking for micro air vehicles in obstacle-rich environments. *2009 AIAA Guidance, Navigation, and Control Conference*. Chicago, IL.
4. Cowling, I., O. Yakimenko, J. Whidborne, and A. Cooke. 2010. Direct method based control system for an autonomous quadrotor. *J. Intelligent Robotic Syst.* 60:285–316. Chicago, IL.
5. Richards, A., and J. How. 2002. Aircraft trajectory planning with collision avoidance using mixed integer linear programming. *2002 American Control Conference Proceedings*.
6. Kuwata, Y., A. Richards, T. Schouwenaars, and J. How. 2007. Distributed robust receding horizon control for multivehicle guidance. *IEEE Transactions on Control Syst. Technol.* 15(4):627–41.
7. Goerzen, C., Z. Kong, and B. Mettler. 2010. A survey of motion planning algorithms from the perspective of autonomous UAV guidance. *J. Intelligent Robotic Syst.* 57:65–100.
8. Bouabdallah, S., P. Murrieri, and R. Siegwart. 2004. Design and control of an indoor micro quadrotor. *Robotics and Automation 2004 ICRA'04, IEEE International Conference Proceedings*. 5:4393–98.
9. Hoffmann, G., H. Huang, S. Waslander, and C. Tomlin. 2007. Quadrotor helicopter flight dynamics and control: Theory and experiment. *AIAA Guidance, Navigation, and Control Conference Proceedings*. 1–20.
10. Kendoul, F., Z. Yu, and K. Nonami. 2010. Guidance and nonlinear control system for autonomous flight of minirotorcraft unmanned aerial vehicles. *J. Field Robotics* 27(3):311–34.
11. Cook, M. 2007. *Flight dynamics principles*. Elsevier aerospace engineering ser. Elsevier/Butterworth-Heinemann.
12. Kim, H., D. Shim, and S. Sastry. 2002. Nonlinear model predictive tracking control for rotorcraft-based unmanned aerial vehicles. *2002 American Control Conference Proceedings*. 5:3576–81.
13. Liu, C., W.-H. Chen, and J. Andrews. 2011. Piecewise constant model predictive control for autonomous helicopters. *Robotics Autonomous Syst.* 59(7-8):571–79.
14. Bellingham, J., A. Richards, and J. How. 2002. Receding horizon control of autonomous aerial vehicles. *2002 American Control Conference Proceedings*. 5:3741–46.
15. Mettler, B., N. Dadkhah, and Z. Kong. 2010. Agile autonomous guidance using spatial value functions. *Control Eng. Practice* 18(7):773–88. Special Issue on Aerial Robotics.

16. Prautzsch, H., W. Boehm, and M. Paluszny. 2002. *Bézier and B-spline techniques*. Springer Verlag.
17. Zerar, M., F. Cazaurang, and A. Zolghadri. 2009. Coupled linear parameter varying and flatness-based approach for space re-entry vehicles guidance. *Control Theory Applications, IET* 3(8):1081–92.
18. Apkarian, P., P. Gahinet, and G. Becker. 1995. Self-scheduled [infinity] control of linear parameter-varying systems: A design example. *Automatica* 31(9):1251–61.
19. Falcone, P., F. Borrelli, H. E. Tseng, J. Asgari, and D. Hrovat. 2008. Linear time-varying model predictive control and its application to active steering systems: Stability analysis and experimental validation. *Int. J. Robust Nonlinear Control* 18(8):862–75.
20. Maciejowski, J. 2002. *Predictive control: With constraints*. Pearson education.