An explicit MPC for quadrotor trajectory tracking

Cunjia Liu[1], Hao Lu[2], Wen-Hua Chen[1]

1. Department of Aeronautical and Automotive Engineering, Loughborough University, UK, LE11 3TU,
E-mail: {c.liu5, w.chen}@lboro.ac.uk

2. School of Instrumentation Science and Optoelectronics Engineering, Beihang University, Beijing, China
E-mail: luhao@aspe.buaa.edu.cn

Abstract: This paper intends to provide an explicit solution of model predictive control (MPC) for trajectory tracking of quadrotors. This kind of MPC can be designed in a conventional way where a nonlinear optimisation problem needs to be solved online. In this paper by using the differential flatness property of the quadrotor, the reference trajectory, system outputs and inputs can be represented using Bezier curves. Thus, the formulated optimisation problem can be parameterised and converted into standard quadratic programming. The resulted quadratic programming problem can be further formulated into multiparametric quadratic programming which is then solved off-line as a piecewise affine function. The involved manipulations including the reduction of the number of optimisation variables and constraints are detailed in this paper. The performance of the proposed control scheme is demonstrated in a simulation study.

Key Words: Model predictive control, multi-parametric programming, flight control, trajectory tracking

1 Introduction

Model Predictive Control (MPC) is a popular control method especially in the area of process control where online optimisation is usually allowed timewise to incorporate the system nonlinearity, constraints and future reference. Despite the associated computational load, its distinguish features have extended its applications to many systems with fast dynamics. In particular, recent years have seen an increasing number of MPC design for flight control of unmanned aerial vehicles (UAVs). These applications usually provide improved control performance but at the cost of increased computational loads or complicated software structures due to the online optimisation.

Various MPC techniques have been developed to fulfil the requirement on the sampling rate when applying MPC to UAVs, which have complicated, usually nonlinear dynamics. The most common approach is though linearisation. Early studies tend to the design MPC based on a single linearised model [1]. A more sophisticated solution is to use multiple linearised models to formulate a switching MPC, which has been used in quadrotor control [2]. More recently, a linear time varying MPC is designed for trajectory tracking of a fixed-wing UAV, where the linearisation is performed along a pre-defined flat trajectory [3]. Nonlinear MPC has also been seen in the literature that was usually combined with other control structures like low-level linear controllers to compensate the associated low sampling rate [4, 5].

Different from above mentioned methods, this paper aims to provide an explicit MPC solution to nonlinear systems of differential flatness. This feature has been exploited in many UAV path planning algorithms (see e.g. [6]). Loosely speaking, the differential flatness suggests that the system's state and input can be expressed by using the output and its higher order derivatives. In designing MPC for such systems, the open-loop finite-time optimal control can be cast into a constrained nonlinear optimisation problem by following normal design procedures. The formulated optimisation problem then needs to be solved at each sampling instant where the calculation time determines the sampling rate. The main contribution of this paper is therefore on the practical side to provide a fast or explicit solution when implementing the corresponding MPC online. The underlying idea is to parameterise the nonlinear dynamics within the optimisation using a finite number of variables so that the optimisation can be solved in a solution space of lower dimensions.

Specifically, by exploiting the differential flatness of the system, Bezier curves are adopted to parametrise the system outputs as well as the system states and inputs in the MPC formulation. A similar idea of using polynomial curves in nonlinear MPC design has been reported in [7]. However, the proposed method shows that the original nonlinear MPC can be converted into a standard quadratic programming (QP) by appropriate manipulations. This in turn can be solved by multi-parametric quadratic programming (mpQP) off-line and implemented online as a piecewise affine function via a lookup table [8]. The proposed explicit MPC avoids online optimisation thus can be easily deployed on simple embedded systems like low-cost autopilots.

The proposed MPC design and the corresponding derivation of explicit solution is demonstrated through a case study on trajectory tracking control of a quadrotor UAV. Although flight control for quadrotors has become increasingly matured in recent years, the proposed MPC scheme for outer-loop tracking design offers a number of unique features towards the quadrotor operation in practice, including the prediction feature and constraint handling, which will be discussed during the design process. The performance of the proposed MPC scheme is demonstrated through a simulation study.

The reminder of this paper is organised as follows. First, the basic dynamics of a quadrotor and its trajectory tracking function are briefly discussed in Section 2. Then, the problem formulation is provided in details in Section 3, which includes the MPC formulation, parameterisation using Bezier curves and transformation into quadratic programming. Section 4 shows that the the formulated quadratic programming problem can be further simplified and solved by the mpQP technique off-line. The simulation study is presented in Section 6 to show the performance of the proposed control scheme following by conclusions in Section 6.

2 Quadrotor trajectory tracking

The dynamics of a quadrotor helicopter have been well studied in literature (see e.g. [9]). To maintain the completeness of this paper, the dynamic equations are listed as follows:

$$m\ddot{\boldsymbol{\xi}} = R(\boldsymbol{\eta})\boldsymbol{\mu} - mg\boldsymbol{e}_3 \tag{1a}$$

$$M(\boldsymbol{\eta})\ddot{\boldsymbol{\eta}} + C(\boldsymbol{\eta}, \dot{\boldsymbol{\eta}}) = \boldsymbol{\nu} \tag{1b}$$

where $\boldsymbol{\xi} = [\ x \quad y \quad z\ ]^T$ denotes the quadrotor position, $\boldsymbol{\eta} = [\ \phi \quad \theta \quad \psi\ ]^T$ denotes the attitude angle and $\boldsymbol{e}_3 = [\ 0 \quad 0 \quad 1\ ]^T$ is an unit vector. The control force vector is defined as $\boldsymbol{\mu} = [\ 0 \quad 0 \quad u\ ]^T$, where u is the total thrust, and $\boldsymbol{\nu} = [\ \nu_\phi \quad \nu_\theta \quad \nu_\psi\ ]^T$ is the control torque vector. $R(\boldsymbol{\eta})$, $M(\boldsymbol{\eta})$ and $C(\boldsymbol{\eta}, \dot{\boldsymbol{\eta}})$ are rotational matrix, pseudoinertia matrix and Coriolis vector matrix, respectively (see [9] for details). It is worth noting that the model (1) is generic for rotorcraft where the analytical expression for the four control inputs $(u, \nu_\phi, \nu_\theta, \nu_\psi)$ depend on the configuration of the vehicle. Therefore, the proposed MPC design can also been applied to traditional unmanned helicopters.

A hierarchical control approach is very common for quadrotor control design. An outer-loop controller focusing on translational dynamics (1a) is designed to track a predefined trajectory by generating the desired attitude command. An inner-loop controller based on (1b) is responsible to achieve the desired attitude by generating appropriate thrusts on each rotor. To this end, this paper focuses on the outer-loop tracking control by using the proposed explicit MPC. As a benefit the MPC scheme is able to limit the desire attitude commands within a reasonable range. Then, the attitude controller can be designed by many classic, including linear, control methods to achieve the overall tracking performance.

To focus the derivation of the MPC scheme, only the longitudinal and lateral dynamics are considered. The compact system equation is expressed as

$$\begin{aligned} \dot{\boldsymbol{x}} &= f(\boldsymbol{x}, \boldsymbol{u}) \\ \boldsymbol{y} &= h(\boldsymbol{x}) \end{aligned} \tag{2}$$

The system output is defined as $\boldsymbol{y} = [\ x \quad y\ ]^T$, the state is denoted as $\boldsymbol{x} = [\ x \quad \dot{x} \quad y \quad \dot{y}\ ]^T$ and the control input is chosen as $\boldsymbol{u} = [\ \phi \quad \theta\ ]^T$. For more general systems, the system dimensions are assumed to be $\boldsymbol{y} \in \mathbb{R}^{n_y}$, $\boldsymbol{x} \in \mathbb{R}^{n_x}$ and $\boldsymbol{u} \in \mathbb{R}^{n_u}$.

The quadrotor dynamics are differentially flat. Hence system (2) can be expressed as

$$\boldsymbol{x} = \mathcal{F}_x(\boldsymbol{y}, \dot{\boldsymbol{y}}, \ldots, \boldsymbol{y}^{[\rho]}) \tag{3}$$

$$\boldsymbol{u} = \mathcal{F}_u(\boldsymbol{y}, \dot{\boldsymbol{y}}, \ldots, \boldsymbol{y}^{[\kappa]}) \tag{4}$$

Regarding the quadrotor dynamics (2), the relation (3) can be obtained immediately, whereas the control input (4) can be specified as

$$\begin{aligned} \phi &= \sin^{-1}\left(\frac{m}{u}(-\ddot{x}\sin\psi + \ddot{y}\cos\psi)\right) \\ \theta &= \tan^{-1}\left(-\frac{\ddot{x}\cos\psi + \ddot{y}\sin\psi}{g - \ddot{z}}\right) \end{aligned} \tag{5}$$

The differential flatness can be used in MPC design to incorporate the system dynamics. Moreover, Eq.(5) will be used to calculate the desired attitude commands for the low-level attitude controller once the output $\boldsymbol{y}$ and its derivatives are calculated [10].

3 Problem Formulation

Exploiting the differential flatness property of the quadrotor allows one to express the quadrotor state in the output space. To represent the system outputs $\boldsymbol{y}$ and its derivatives, Bezier curves are adopted in this paper. The unique features associated with Bezier curves are able to facilitate the derivation of the explicit MPC.

3.1 MPC formulation

The basic implementation process of MPC is briefly described as follows. At the each sampling time t, the finite-time optimal control over the prediction horizon T is formulated as an optimisation problem. It is then solved online to obtain the control sequence $\boldsymbol{u}(\tilde{\tau})$, $\tilde{\tau} \in [t, t+T]$, where $\tilde{\tau}$ is used to represent the prediction time to distinguish from the real time t. Only a small portion of the resulted control signals will be applied to the actual system until the next sampling instant when the new measurement is arrived. Then, the above procedure is repeated.

A general objective function for finite-time optimal control can be designed to penalise the tracking errors and the control efforts while taking into account the system dynamics and input constraints. Specifically, it can be formulated as

$$\begin{aligned} J(t) =& \|\boldsymbol{w}(t+T) - \boldsymbol{y}(t+T)\|_P^2 + \\ & \int_0^T \|\boldsymbol{w}(t+\tilde{\tau}) - \boldsymbol{y}(t+\tilde{\tau})\|_Q^2 + \|\boldsymbol{u}(t+\tilde{\tau})\|_R^2 \,\mathrm{d}\tilde{\tau} \end{aligned} \tag{6}$$

where $\boldsymbol{w} = [\ x_r(t) \quad y_r(t)\ ]^T$ is the reference trajectory, $\boldsymbol{y} = [\ x(t) \quad y(t)\ ]^T$ denotes the system outputs, and $\boldsymbol{u}$ is the control inputs. $\|\boldsymbol{x}\|_A = \boldsymbol{x}^T A \boldsymbol{x}$ denotes the weighted euclidean norm of $\boldsymbol{x}$, and P, Q and R are positive-definite weighting matrices for terminal penalty, process cost and input cost, respectively. For the sake of simplicity they can be specified as $P = \mathrm{diag}\{p_1, p_2\}$, $Q = \mathrm{diag}\{q_1, q_2\}$ and $R = \mathrm{diag}\{r_1, r_2\}$, respectively.

The optimisation problem for MPC can be formulated as

$$\min_{\boldsymbol{u}(\tilde{\tau})} J(t) \tag{7}$$

subject to

$$\begin{aligned} \dot{\boldsymbol{x}} &= \mathbf{f}(\boldsymbol{x}, \boldsymbol{u}) \\ \boldsymbol{x}_0 &= \boldsymbol{x}(t) \\ \boldsymbol{u}_{min} &\le \boldsymbol{u} \le \boldsymbol{u}_{max} \end{aligned} \tag{8}$$

where $\boldsymbol{x}_0 = [\ x_0 \quad y_0 \quad \dot{x}_0 \quad \dot{y}_0\ ]^T$ is the initial state including position and velocity, $\boldsymbol{u}_{min}$ and $\boldsymbol{u}_{max}$ are the lower bound and upper bound for the accelerations, respectively.

To find the optimal $\boldsymbol{u}(\tilde{\tau})$, for $\tilde{\tau} \in [0, T]$ discretisation or parameterisation needs to be performed to transfer this optimisation problem into a nonlinear programming problem which depends on a finite number of optimisation variables.

In this paper, Bezier curves are used to parameterise the system outputs as well as the system states and inputs which all depend on the coefficients of the Bezier curves (also known as control points).

3.2 Parametrisation using Bezier curve

Different from the common polynomial form of Bezier curve, this paper advocates the matrix form expression because it will facilitate the derivation of the explicit MPC solution. In general, the n-th order Bezier curves can be expressed in a matrix form, such that

$$B_n(\tau) = \boldsymbol{\tau} V \boldsymbol{\lambda}, \tag{9}$$

where $\boldsymbol{\tau}$ is the polynomial basis vector defined by time parameter τ, such that

$$\boldsymbol{\tau}(\tau) = \begin{bmatrix} 1 & \tau & \cdots & \tau^n \end{bmatrix}, \tag{10}$$

V is a weight matrix of lower-triangular form defined as

$$V = \begin{bmatrix} v_{(1,1)} & \cdots & 0 \\ \vdots & \ddots & \vdots \\ v_{(n+1,1)} & \cdots & v_{(n+1,n+1)} \end{bmatrix} \tag{11}$$

with its elements defined by $v_{(i,j)} = \frac{n!(-1)^{i-j}}{(j-1)!(i-j)!(n-i+1)!}$ for $i > j$, and $\boldsymbol{\lambda}$ is the control point vector such that

$$\boldsymbol{\lambda} = \begin{bmatrix} \lambda_0 & \cdots & \lambda_n \end{bmatrix}^T \tag{12}$$

which determines the shape of the Bezier curve. Since the time parameter of the Bezier curve τ is usually defined over an interval $\tau \in [0, 1]$, for a trajectory defined in real-time $t \in [0, T]$ the mapping can be achieved by setting $\tau = t/T$.

One feature of using matrix form for Bezier curve is that the derivatives and integral of the Bezier curve $B_n(\tau)$ with respect to the time parameter τ can be easily calculated by applying the corresponding operations only on the polynomial basis vector $\boldsymbol{\tau}$.

By exploiting the Bezier curve defined in (9), the output of the quadrotor system can be expressed as follows:

$$\boldsymbol{y}(t) = \begin{bmatrix} B_{x,n}(t/T) \\ B_{y,n}(t/T) \end{bmatrix} = \underbrace{\begin{bmatrix} \boldsymbol{\tau}(\frac{t}{T}) & \\ & \boldsymbol{\tau}(\frac{t}{T}) \end{bmatrix}}_{\bar{\boldsymbol{\tau}}(t)} \underbrace{\begin{bmatrix} V & \\ & V \end{bmatrix}}_{\bar{\mathbf{V}}} \underbrace{\begin{bmatrix} \boldsymbol{\lambda}_x \\ \boldsymbol{\lambda}_y \end{bmatrix}}_{\bar{\boldsymbol{\lambda}}} \tag{13}$$

where $\boldsymbol{\lambda}_x$ and $\boldsymbol{\lambda}_y$ are control point vectors for x direction and y direction, respectively.

The control input in MPC design is chosen to be the translational acceleration, which can be converted into attitude angle via (5). Its expression can be derived from (13) by performing differentiation such that

$$\boldsymbol{u}(t) = \boldsymbol{y}^{[2]}(t) = \underbrace{\begin{bmatrix} \boldsymbol{\tau}_u(t) & \\ & \boldsymbol{\tau}_u(t) \end{bmatrix}}_{\bar{\boldsymbol{\tau}}_u(t)} \bar{\mathbf{V}} \bar{\boldsymbol{\lambda}} \tag{14}$$

where

$$\boldsymbol{\tau}_u(t) = \begin{bmatrix} 0 & 0 & \frac{2}{T^2} & \cdots & \frac{n(n-1)t^{n-2}}{T^n} \end{bmatrix}. \tag{15}$$

Note that the constraints imposed on the translational acceleration can also limit the attitude angles.

The reference trajectory $\boldsymbol{w}(t)$ can also be represented by Bezier curves. This can be easily achieved by various path planning or smoothing algorithms. In this work, it is assumed the reference is a m-th order Bezier curve defined as

$$\boldsymbol{w}(t) = \bar{\boldsymbol{\tau}}_w(t) \bar{\mathbf{V}}_w \bar{\boldsymbol{\lambda}}_w \tag{16}$$

where, matrices $\bar{\boldsymbol{\tau}}_w(t) = \mathrm{diag}\{\boldsymbol{\tau}_w(t), \boldsymbol{\tau}_w(t)\}$ and $\bar{\mathbf{V}}_w = \mathrm{diag}\{V_w, V_w\}$ are defined analogous to (13) with appropriate dimensions and $\bar{\boldsymbol{\lambda}}_w$ is the corresponding control point vector. The parametrisation (16) is normally defined for the entire trajectory from 0 to t_f. To account for the reference trajectory only belongs the prediction horizon $t \le \tilde{\tau} \le t+T$, further manipulations are required.

Consider a general Bezier curve (9) defined over $\tau \in [0, 1]$. A sub-section of this Bezier curve within the interval $\tau \in [a, b]$, $0 \le a < b \le 1$, is also a Bezier curve. The new Bezier curve can be obtained by replacing the original time variable as $\tau = a + \tilde{\tau}(b-a)$, where $\tilde{\tau} \in [0, 1]$. Substituting this relation into (9) gives the sub-section curve

$$\begin{aligned} B'_n(\tilde{\tau}) &= \boldsymbol{\tau}(a + \tilde{\tau}(b-a)) V \boldsymbol{\lambda} \\ &= \boldsymbol{\tau}(\tilde{\tau}) Z V \boldsymbol{\lambda} \\ &= \boldsymbol{\tau}(\tilde{\tau}) V \underbrace{(V^{-1} Z V \boldsymbol{\lambda})}_{\boldsymbol{\lambda}'} \end{aligned} \tag{17}$$

where $\boldsymbol{\lambda}'$ is the new control point vector of the sub-section Bezier curve and the splitting matrix Z is an upper triangular matrix defined as

$$Z = \begin{bmatrix} z_{(1,1)} & \cdots & z_{(1,n+1)} \\ \vdots & \ddots & \vdots \\ 0 & \cdots & z_{(n+1,n+1)} \end{bmatrix} \tag{18}$$

with

$$z_{(i,j)} = \frac{(j-1)!}{(i-1)!(j-i)!} a^{j-i} (b-a)^{i-1}, \quad i \le j. \tag{19}$$

To derive the control point vector of the reference Bezier curve spanning the prediction horizon at each sampling instant t, the interval can be set as $a = t/t_f$ and $b = (t+T)/t_f$. Thus, analogous to system output (13), the corresponding reference curve can be expressed as

$$\boldsymbol{w}(\tilde{\tau}) = \bar{\boldsymbol{\tau}}_w(\tilde{\tau}) \bar{\mathbf{V}}_w \underbrace{(\bar{\mathbf{V}}_w^{-1} \bar{\mathbf{Z}}_w \bar{\mathbf{V}}_w \bar{\boldsymbol{\lambda}}_w)}_{\bar{\boldsymbol{\Lambda}}_0} \tag{20}$$

where $\bar{\mathbf{Z}}_w = \mathrm{diag}\{Z_w, Z_w\}$ and $\bar{\boldsymbol{\Lambda}}_0$ is the control point vector that determines the shape of the reference trajectory within the prediction horizon. Note that the reason of defining a new control point vector for the reference is to guarantee the weight matrix in the optimisation problem to retain a constant so that the mpQP technique can be applied.

3.3 Quadratic programming problem

By using the Bezier curve parametrisation (13) and (20) for the outputs and the reference signals, respectively, the cost function (6) can be rewritten as

$$\begin{aligned} J =& \bar{\boldsymbol{\lambda}}^T \bar{\mathbf{V}}^T (\bar{\mathcal{T}}_P + \bar{\mathcal{T}}_Q + \bar{\mathcal{T}}_R) \bar{\mathbf{V}} \bar{\boldsymbol{\lambda}} \\ & - 2 \bar{\boldsymbol{\Lambda}}_0^T \bar{\mathbf{V}}_w^T (\bar{\mathcal{T}}'_P + \bar{\mathcal{T}}'_Q) \bar{\mathbf{V}} \bar{\boldsymbol{\lambda}} \\ & + \bar{\boldsymbol{\Lambda}}_0^T \bar{\mathbf{V}}_w^T (\bar{\mathcal{T}}_{w,P} + \bar{\mathcal{T}}_{w,Q}) \bar{\mathbf{V}} \bar{\boldsymbol{\Lambda}}_0 \end{aligned} \tag{21}$$

where

$$\bar{\mathcal{T}}_p = \mathrm{diag}\{p_1\mathcal{T}_P, p_2\mathcal{T}_P\}, \quad \mathcal{T}_P = \boldsymbol{\tau}(t)^T\boldsymbol{\tau}(t)|_{t=T} \tag{22}$$

$$\bar{\mathcal{T}}_Q = \mathrm{diag}\{q_1\mathcal{T}_Q, q_2\mathcal{T}_Q\}, \quad \mathcal{T}_Q = \int_0^T \bar{\boldsymbol{\tau}}(t)^T\bar{\boldsymbol{\tau}}(t)\,\mathrm{d}t \tag{23}$$

$$\bar{\mathcal{T}}_R = \mathrm{diag}\{r_1\mathcal{T}_R, r_2\mathcal{T}_R\}, \quad \mathcal{T}_R = \int_0^T \boldsymbol{\tau}_u(t)^T\boldsymbol{\tau}_u(t)\,\mathrm{d}t \tag{24}$$

are constant weighting matrices. By solving the corresponding equations, their analytical solutions can be obtained. Similarly, matrices $\bar{\mathcal{T}}_P'$ and $\bar{\mathcal{T}}_Q'$ can also be constructed with appropriate dimensions. The last term in (21) is a constant thus can be ignored in optimisation.

Given (21), the original nonlinear optimisation problem in (7) is therefore equivalent to

$$\min_{\bar{\boldsymbol{\lambda}}} J = \bar{\boldsymbol{\lambda}}^T\bar{\mathbf{H}}\bar{\boldsymbol{\lambda}} - 2\bar{\boldsymbol{\Lambda}}_0^T\bar{\mathbf{F}}\bar{\boldsymbol{\lambda}} \tag{25}$$

subject to

$$\mathbf{A}\bar{\boldsymbol{\lambda}} \leq \mathbf{B} \tag{26a}$$

$$\mathbf{A}_{\text{eq}}\bar{\boldsymbol{\lambda}} = \mathbf{B}_{\text{eq}} \tag{26b}$$

where $\bar{\mathbf{H}} = \bar{\mathbf{V}}^T(\mathcal{T}_P + \mathcal{T}_Q + \mathcal{T}_R)\bar{\mathbf{V}}$, $\bar{\mathbf{F}} = \bar{\mathbf{V}}_w^T(\mathcal{T}_P + \mathcal{T}_Q)\bar{\mathbf{V}}$. The inequality constraints are used to limit the control efforts over the predictive horizon, such that

$$\boldsymbol{u}_{min} \leq \bar{\boldsymbol{\tau}}_u(\tilde{\tau})\bar{\mathbf{V}}\bar{\boldsymbol{\lambda}} \leq \boldsymbol{u}_{max}, \quad \tilde{\tau} \in [0, T] \tag{27}$$

To enforce this constraint using a finite number of optimisation variables, define a series of points $0 = t_1 \leq t_i \leq t_s = T$, $i = 1, \ldots, s$, at which the inequality constraints are imposed. The constraint requirement in (25) can be expressed as

$$\mathbf{A} = \begin{bmatrix} \bar{\boldsymbol{\tau}}_u(t_1)\bar{\mathbf{V}} \\ \vdots \\ \bar{\boldsymbol{\tau}}_u(t_p)\bar{\mathbf{V}} \\ -\bar{\boldsymbol{\tau}}_u(t_1)\bar{\mathbf{V}} \\ \vdots \\ -\bar{\boldsymbol{\tau}}_u(t_s)\bar{\mathbf{V}} \end{bmatrix} \quad \mathbf{B} = \begin{bmatrix} \boldsymbol{u}_{max} \\ \vdots \\ \boldsymbol{u}_{max} \\ -\boldsymbol{u}_{min} \\ \vdots \\ -\boldsymbol{u}_{min} \end{bmatrix} \tag{28}$$

The equality constraints are used to enforce the initial system state $\boldsymbol{x}_0$, thus

$$\mathbf{A}_{\text{eq}} = \begin{bmatrix} \bar{\boldsymbol{\tau}}(0)\bar{\mathbf{V}} \\ \bar{\boldsymbol{\tau}}_v(0)\bar{\mathbf{V}} \end{bmatrix} \quad \mathbf{B}_{\text{eq}} = \boldsymbol{x}_0 \tag{29}$$

where $\bar{\boldsymbol{\tau}}_v(t) = \bar{\boldsymbol{\tau}}^{[1]}(t) = \begin{bmatrix} 0 & \frac{1}{T} & \frac{2t}{T^2} & \cdots & \frac{nt^{(n-1)}}{T^n} \end{bmatrix}$.

At this point, the nonlinear optimisation problem involved in the NMPC design is converted into a quadratic programming, which can be solved by standard QP solvers. For some applications with relatively powerful onboard computer, this problem can be solve by online solvers. However, the some low-cost flight computer, an explicit solution is still very appealing. To this end, the next section shows the procedure to further simplify the QP Problem (25) and solve it off-line using multi-parameter programming.

4 Multi-parametric programming

For the formulated QP problem (25), a solution needs to be found at each sampling instant given a new measurement $\boldsymbol{x}_0$ and reference parameter $\bar{\boldsymbol{\Lambda}}_0$. Instead of using a numerical QP solver, the solution can be pre-calculated off-line for the entire set of state $\boldsymbol{x}$ and reference $\boldsymbol{w}$ of interest by using the multiparametric quadratic programming. Such an explicit solution can be retrieved online by evaluating a piecewise affine optimiser function. A comprehensive review about this method can be found in [11].

The mpQP algorithm utilises the Karush-Kuhn-Tucker (KKT) conditions for optimality to construct the solution. The complexity of the solution depends on the number of constraints and parameters. To reduce the number of constraints, especially the equality constraints (26b), the initial state $\boldsymbol{x}_0$ can be used to directly determine the first few control points in a Bezier curve. Solving the equality constraint (26b) gives

$$\begin{bmatrix} \lambda_{x,0} \\ \lambda_{x,1} \end{bmatrix} = \begin{bmatrix} v_{(1,1)} & 0 \\ v_{(2,1)} & v_{(2,2)} \end{bmatrix}^{-1} \begin{bmatrix} 1 & 0 \\ 0 & 1/T \end{bmatrix}^{-1} \begin{bmatrix} x_0 \\ \dot{x}_0 \end{bmatrix} \tag{30}$$

Similarly, control points $(\lambda_{y,0}, \lambda_{y,1})$ can also be obtained via $(y_0, \dot{y}_0)$

To separate the known control points from the unknown ones within $\bar{\boldsymbol{\lambda}}$ thus to simplify the optimisation problem, partitioning the relevant matrices and vectors is required. Without losing generality, assume the number of known control points of one Bezeir cure is n_r, then the weight matrix $H = V^T(\mathcal{T}_P + \mathcal{T}_Q + \mathcal{T}_R)V$ can be rewritten as

$$H = \begin{bmatrix} H_{11} & H_{12} \\ H_{21} & H_{22} \end{bmatrix} \in \mathcal{R}^{(n+1)\times(n+1)} \tag{31}$$

where $H_{11} \in \mathcal{R}^{n_r \times n_r}$ such that H_{12}, H_{21} and H_{22} are matrices with appropriate dimensions. Similarly, the weight matrix V can be partitioned into

$$V = \begin{bmatrix} V_1 & V_2 \end{bmatrix} \in \mathcal{R}^{(n+1)\times(n+1)} \tag{32}$$

where $V_1 \in \mathcal{R}^{(n+1)\times n_r}$ such that V_2 is a matrix with appropriate dimension.

Redefining the optimisation variables by rearranging the control point vector $\bar{\boldsymbol{\lambda}}$ gives

$$\boldsymbol{\Lambda}_1 = \begin{bmatrix} \lambda_{x,0} & \lambda_{x,1} & \lambda_{y,0} & \lambda_{y,1} \end{bmatrix}^T \tag{33}$$

and

$$\boldsymbol{\Lambda}_2 = \begin{bmatrix} \lambda_{x,2} & \cdots & \lambda_{x,n} & \lambda_{y,2} & \cdots & \lambda_{y,n} \end{bmatrix}^T \tag{34}$$

where $\boldsymbol{\Lambda}_1$ denotes for the known control points and $\boldsymbol{\Lambda}_2$ denotes for the unknown control points which need to be found by optimisation.

The cost function can be converted to

$$\begin{aligned} J =& \boldsymbol{\Lambda}_2^T\bar{\mathbf{H}}_{22}\boldsymbol{\Lambda}_2 + 2(\boldsymbol{\Lambda}_1^T\bar{\mathbf{H}}_{12} - \boldsymbol{\Lambda}_0^T\bar{\mathbf{F}}_2)\boldsymbol{\Lambda}_2 + \\ & \boldsymbol{\Lambda}_1^T\bar{\mathbf{H}}_{11}\boldsymbol{\Lambda}_1 - 2\boldsymbol{\Lambda}_0^T\bar{\mathbf{F}}_1\boldsymbol{\Lambda}_1 \end{aligned} \tag{35}$$

where $\bar{\mathbf{H}}_{22} = \mathrm{diag}\{H_{22}, H_{22}\}$, $\bar{\mathbf{H}}_{12} = \mathrm{diag}\{H_{12}, H_{12}\}$, $\bar{\mathbf{F}}_1 = \mathrm{diag}\{F_1, F_1\}$ and $\bar{\mathbf{F}}_2 = \mathrm{diag}\{F_2, F_2\}$

To Define parameter $\mathbf{\Theta} = \begin{bmatrix}\mathbf{\Lambda}_0^T & \mathbf{\Lambda}_1^T\end{bmatrix}^T$. Converted QP can be formulated as

$$\begin{aligned}\min_{\mathbf{\Lambda}_2} J =& \mathbf{\Lambda}_2^T \bar{\mathbf{H}}_{22} \mathbf{\Lambda}_2 + 2(\mathbf{\Theta}^T \mathbf{F}_\Theta)\mathbf{\Lambda}_2 \\ \textbf{s.t.} \quad & \mathbf{A}_\Theta \mathbf{\Lambda}_2 \leq \mathbf{B}_\Theta \mathbf{\Theta} + \mathbf{B}\end{aligned} \tag{36}$$

where $\mathbf{F}_\Theta = \begin{bmatrix} -\bar{\mathbf{F}}_2^T & \bar{\mathbf{H}}_{12}^T \end{bmatrix}^T$, $\mathbf{A}_\Theta$ and $\mathbf{B}_\Theta$ are defined in the following formats

$$\mathbf{A}_\Theta = \begin{bmatrix} A_{(1)}^T & \cdots & A_{(s)}^T & -A_{(1)}^T & \cdots & -A_{(s)}^T \end{bmatrix}^T \tag{37}$$

$$\mathbf{B}_\Theta = \begin{bmatrix} B_{(1)}^T & \cdots & B_{(s)}^T & -B_{(1)}^T & \cdots & -B_{(s)}^T \end{bmatrix}^T \tag{38}$$

respectively. Their individual entities are defined as $A_{(i)} = \bar{\boldsymbol{\tau}}_u(t_i)\bar{\mathbf{V}}_2$ and $B_{(i)} = -\bar{\boldsymbol{\tau}}_u(t_i)\bar{\mathbf{V}}_1, \forall i \in \{1, \ldots, s\}$, where

$$\bar{\mathbf{V}}_1 = \mathrm{diag}\{V_1, V_1\} \tag{39}$$

$$\bar{\mathbf{V}}_2 = \mathrm{diag}\{V_2, V_2\} \tag{40}$$

The formulated mpQP problem (36) has constant weight matrices in the cost function and a parameter vector to represent the reference and initial state. Therefore, it can be solved by mpQP techniques, e.g. the Multi-Parametric Toolbox [12], and incorporated into embedded hardware for fast online implementation.

5 Simulation study

The simulation study is based on a full quadrotor model with both translational dynamics and attitude dynamics. The proposed explicit MPC is used to develop a trajectory tracking controller, which is able to take the reference defined by Bezier curves and provide attitude commands within a suitable range. The low-level controller, including the attitude control and the high-heading control, is designed by using the proportional-derivative (PD) approach, hence is not presented here.

The MPC for trajectory tracking is designed with the weight matrices $P = \mathrm{diag}\{2, 2\}$, $Q = \mathrm{diag}\{1, 1\}$ and $R = \mathrm{diag}\{2, 2\}$. The prediction horizon is set to $T = 2$ seconds. The input constraints are imposed on the system accelerations, such that they fall into the range $[-3\mathrm{m/s}^2, 3\mathrm{m/s}^2]$. The proposed MPC assumes that the reference is constructed by cubic Bezier curves, i.e. $m = 3$, since they are very popular in UAV path planning [13]. The order of Bezier curves used in the MPC design is set to $n = 4$. Considering the number of known control points from the initial state, the dimensions of the parameter vector and optimisation variables in mpQP are $\mathbf{\Theta} \in \mathcal{R}^{12}$ and $\mathbf{\Lambda}_2 \in \mathcal{R}^6$, respectively. The constraints checking points is set to $s = 5$, therefore the number of constraints is 20. The region of the parameter vector $\mathbf{\Theta}$ also needs to be specified. In this work it is determined by the operational range of the quadrotor, such that the control points are within a square box of $2\mathrm{km}$ side length centred in the origin.

The formulated mpQP problem is solved by using Multi-Parametric Toolbox 3.0 [12], which results in 2203 parameter regions. An illustration of a subspace partition is given in Fig. 1, where all the parameters are fixed at zeros except for $\lambda_{x,1}$ and $\lambda_{y,1}$. This corresponds to the scenario of hovering at the origin with different initial velocities. The piecewise affine solution with this size can be easily stored and implemented online. This also shows the potential for expanding the design to three dimensional tracking design. Given this explicit solution, the sampling rate can easily reach to 50Hz in the MPC implementation .

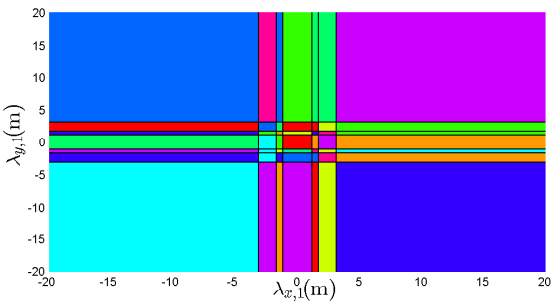

Fig. 1: Partition illustration

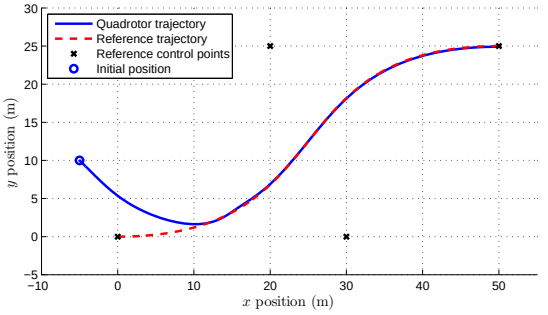

Fig. 2: Quadrotor trajectory tracking result

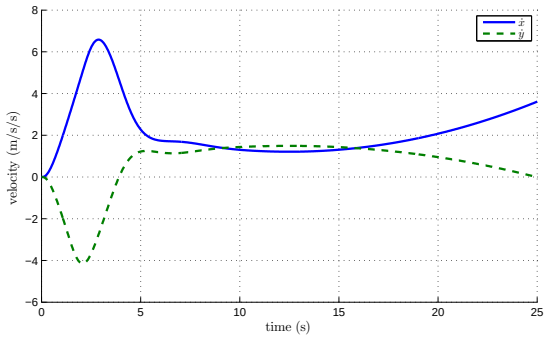

Fig. 3: Quadrotor velocity profile

The trajectory tracking result from the simulation is shown in Fig. 2. The reference trajectory is a cubic Bezier curve defined by four control points, which is a manoeuvre usually used to connect two parallel straight-lines. The total time for completing this track is 25 seconds. As the kinematic model is adopted in the MPC design without any linearisation, the initial quadrotor position is set far away from the reference trajectory to demonstrate the ability of handling the nonlinearity. It can be seen from Fig. 2 that the quadrotor is guided back to the reference trajectory

smoothly. The corresponding velocity profile is given in Fig. 3. Another feature of the MPC is the constraint handling. To this end, the attitude commands generated by the proposed MPC, including the roll angle and the pitch angle, are presented in Fig. 4 and Fig. 5, respectively. Note that by limiting the horizontal acceleration to $\pm 3\mathrm{m}/\mathrm{s}^2$, both attitude commands are within $\pm 0.3\mathrm{rad}$. It can been observed that although an aggressive manoeuvre is required at the beginning of the tracking, the attitude angles are well maintained within a safe range. The relation between the attitude motion and the horizontal acceleration has also been demonstrated in this simulation. On the other hand, with the help of the low-level controller the quadrotor is able to follow these attitude commands so as to delivered a satisfactory trajectory tracking performance.

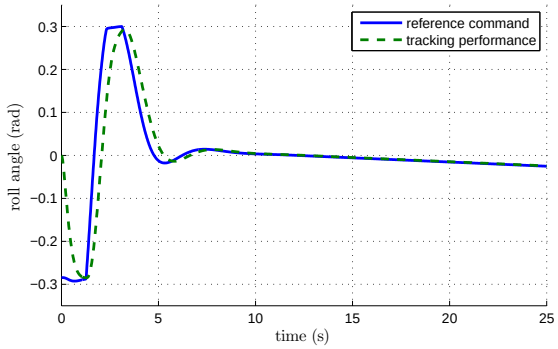

Fig. 4: Roll angle command and tracking performance

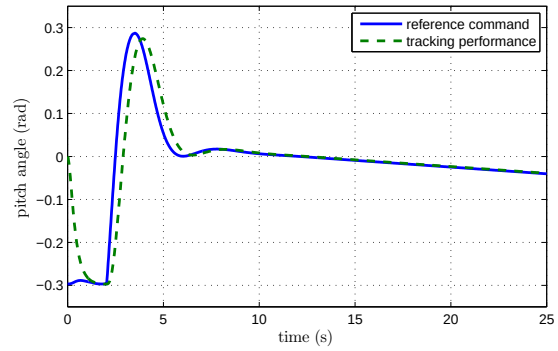

Fig. 5: Pitch angle command and tracking performance

6 Conclusions

This paper describes an explicit MPC solution for quadrotor trajectory tracking. The controller is designed to track a trajectory defined by Bezier curves and to provide attitude commands for a low-level attitude controller. In the MPC design, the vehicle's acceleration as well as the attitude command can be constrained within a reasonable range thus to reduce the operational risk. Using the differential flatness property allows the optimisation problem in the formulated tracking MPC to be parameterised by Bezier curves. Therefore, it can be converted into a quadratic programming problem and solved off-line using mpQP techniques. The implementation thus becomes a simple evaluation of a piecewise affine function. The simulation study shows that the complexity of the explicit solution is well controlled and the proposed controller is able to deliver satisfactory tracking performance. The proposed explicit MPC also has the potentials of being applied on similar systems with differential flatness where fast online implementation is required.

References

[1] C.L. Castillo, W. Moreno, and K.P. Valavanis, "Unmanned helicopter waypoint trajectory tracking using model predictive control," in *Control Automation, 2007. MED '07. Mediterranean Conference on*, June 2007, pp. 1–8.

[2] Kostas Alexis, George Nikolakopoulos, and Anthony Tzes, "Switching model predictive attitude control for a quadrotor helicopter subject to atmospheric disturbances," *Control Engineering Practice*, vol. 19, no. 10, pp. 1195 – 1207, 2011.

[3] Ionela Prodan, Sorin Olaru, Ricardo Bencatel, Joo Borges de Sousa, Cristina Stoica, and Silviu-Iulian Niculescu, "Receding horizon flight control for trajectory tracking of autonomous aerial vehicles," *Control Engineering Practice*, vol. 21, no. 10, pp. 1334 – 1349, 2013.

[4] D.H. Shim, H.J. Kim, and S. Sastry, "Decentralized nonlinear model predictive control of multiple flying robots," in *Decision and Control, 2003. Proceedings. 42nd IEEE Conference on*, Dec 2003, vol. 4, pp. 3621–3626 vol.4.

[5] Cunjia Liu, Wen-Hua Chen, and John Andrews, "Piecewise constant model predictive control for autonomous helicopters," *Robotics and Autonomous Systems*, vol. 59, no. 78, pp. 571 – 579, 2011.

[6] Ian D. Cowling, Oleg A. Yakimenko, James F. Whidborne, and Alastair K. Cooke, "Direct method based control system for an autonomous quadrotor," *Journal of Intelligent & Robotic Systems*, vol. 60, no. 2, pp. 285–316, 2010.

[7] Radhakrishnan Mahadevan, Sunil K. Agrawal, and Francis J.Doyle III, "Differential flatness based nonlinear predictive control of fed-batch bioreactors," *Control Engineering Practice*, vol. 9, no. 8, pp. 889–899, 2001, Advanced Control of Chemical Processes.

[8] Alberto Bemporad, Manfred Morari, Vivek Dua, and Efstratios N. Pistikopoulos, "The explicit linear quadratic regulator for constrained systems," *Automatica*, vol. 38, no. 1, pp. 3 – 20, 2002.

[9] Farid Kendoul, David Lara, Isabelle Fantoni, and R Lozano, "Real-time nonlinear embedded control for an autonomous quadrotor helicopter," *Journal of guidance, control, and dynamics*, vol. 30, no. 4, pp. 1049–1061, 2007.

[10] Cunjia Liu and Wen-Hua Chen, "A practical receding horizon control framework for path planning and control of autonomous VTOL vehicles," in *EUCASS Proceedings Series Advances in AeroSpace Sciences*, 2013, vol. 6, pp. 159–174.

[11] Alessandro Alessio and Alberto Bemporad, "A survey on explicit model predictive control," in *Nonlinear Model Predictive Control*, Lalo Magni, Davide Martino Raimondo, and Frank Allgower, Eds., vol. 384 of *Lecture Notes in Control and Information Sciences*, pp. 345–369. Springer Berlin Heidelberg, 2009.

[12] M. Herceg, M. Kvasnica, C.N. Jones, and M. Morari, "Multi-parametric toolbox 3.0," in *Control Conference (ECC), 2013 European*, July 2013, pp. 502–510.

[13] Kwangjin Yang and S. Sukkarieh, "3d smooth path planning for a uav in cluttered natural environments," in *Intelligent Robots and Systems, 2008. IROS 2008. IEEE/RSJ International Conference on*, Sept 2008, pp. 794–800.