2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
Madrid, Spain, October 1-5, 2018

Flatness-based Model Predictive Control for Quadrotor Trajectory Tracking

Melissa Greeff and Angela P. Schoellig

***Abstract*— The use of model predictive control for quadrotor applications requires balancing trajectory tracking performance and constraint satisfaction with fast computation. This paper proposes a Flatness-based Model Predictive Control (FMPC) approach that can be applied to quadrotors, and more generally, differentially flat nonlinear systems. Our proposed FMPC couples feedback model predictive control with feedforward linearization. The proposed approach has the computational advantage that, similar to linear model predictive control, it only requires solving a convex quadratic program instead of a nonlinear program. However, unlike linear model predictive control, we still account for the nonlinearity in the model through the use of an inverse term. In simulation, we demonstrate improved robustness over approaches that couple model predictive control with feedback linearization. In experiments using quadrotor vehicles, we also demonstrate improved trajectory tracking compared to classical linear and nonlinear model predictive control approaches.**

I. INTRODUCTION

The growing interest in unmanned aerial vehicles (UAVs) for applications such as infrastructure inspection [1], search-and-rescue missions [2] and mapping operations [3] has challenged researchers to develop controllers that can move beyond lab demonstrations to real-world scenarios. Successful controllers therefore must meet the following three criteria: exhibit high-trajectory tracking performance; explicitly account for input and state constraints; and demonstrate robustness to unmodelled dynamics, disturbances and time delays. Furthermore, the inherently fast dynamics of UAVs require real-time operation, on-board, in a high-frequency feedback loop.

Model predictive control (MPC) is a popular approach to meet the first two criteria by optimizing over a prediction horizon while still explicitly adhering to constraints on the states and inputs of the system [4]-[6]. However, the practical challenge is to balance this with the real-time computation requirement.

The most common real-time MPC approach is a *direct* method, which transforms the open-loop optimal control problem into a finite-dimensional problem by first discretizing the model [8]. This model-based controller then generally considers one of three model classes.

Nonlinear Model Predictive Control (NMPC) uses a nonlinear system model which, coupled with *direct* methods,

The authors are with the Dynamic Systems Lab (www.dynsyslab.org) at the University of Toronto Institute for Aerospace Studies (UTIAS), Canada. Email: melissa.greeff@mail.utoronto.ca, schoellig@utias.utoronto.ca

This work was supported by Drone Delivery Canada, Defence Research and Development Canada and Natural Sciences, and Engineering Research Council of Canada.

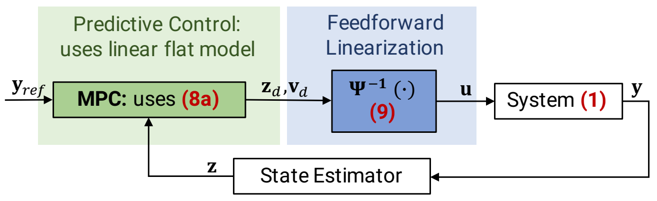


Fig. 1. Overall architecture diagram of the proposed Flatness-based Model Predictive Control (FMPC) approach.

results in solving a non-convex nonlinear program (NLP) at each time step. Current real-time NMPC tends to find a suboptimal solution of the NLP by performing often only one iteration of a sequential quadratic program (SQP) with Gauss-Newton approximation [7], [9]. This is often combined with *warm-starting*, i.e., initializing using the estimate from the previous time step [8]. For longer time horizons, efficiency can be improved by using a *multiple shooting* method, which considers both the system state and input as optimization variables and adds the system dynamics as equality constraints. When solving the NLP, the SQP solver can then exploit the resulting sparse structure of the problem [7].

Alternatively, Linear Model Predictive Control (LMPC) uses a linearized model (often about hover for quadrotors [4]), which, coupled with *direct* methods, results in a convex quadratic program (QP) that can be efficiently solved at each time step.

The final approach, Model Predictive Control combined with Feedback Linearization (MPC+FBL), combines feedback linearization to cancel nonlinear terms with MPC that considers a linear model [10], [11]. This idea was first presented in the mid 1990s. It was shown that using a representative, but modified cost function, can result in a convex QP as in LMPC. MPC in our proposed approach in Fig. 1 similarly solves such a convex QP. Initial work looked promising as simulations showed comparable performance to NMPC but with decreased computational cost [10]. However, the practical implementation of MPC+FBL appears to be stunted by both robustness issues [12] and the required input constraint conversion (see Section IV) [10].

This paper tackles the following question: *Can we make the idea of MPC+FBL, where a linearization term is coupled with linear MPC, practical and implementable?*

To answer this question, we first consider when and how feedback linearization can be applied. Many physical systems, including cranes, cars with trailers and quadrotors, can be described by nonlinear models exhibiting a property known as differential flatness [13], [14]. Intuitively, differential flatness allows us to separate the nonlinear model into a linear dynamics component and a nonlinear transformation. This property can be utilized in both feedback and feedforward linearization [15].

Feedforward linearization aims to overcome the robustness issues of feedback linearization, which may be the result of parametric model uncertainty leading to inexact pole-zero cancellation [15]. In [16], feedforward linearization achieved improved tracking performance over feedback linearization for a ball-plate experiment. Given its robustness advantages, our proposed approach, shown in Fig. 1, couples feedforward linearization with MPC.

The contributions of this paper are three-fold. Firstly, we propose a novel Flatness-based Model Predictive Control (FMPC) architecture that couples feedback MPC with feedforward linearization. Practical advantages (in particular, an ability to account for known input delays and improved robustness to model parameter uncertainty) over MPC+FBL are demonstrated in simulation in Section VI. Secondly, we implement our FMPC architecture on a quadrotor UAV, accounting for inner-loop dynamics and known input time delays. Finally, in Section VII-C we demonstrate promising results for FMPC as an outer-loop controller of a quadrotor UAV with improved trajectory tracking performance over NMPC and LMPC.

II. PROBLEM STATEMENT

Consider a system with a continuous-time, nonlinear model of the form:

$$\begin{aligned} \dot{\mathbf{x}}(t) &= f(\mathbf{x}(t), \mathbf{u}(t)), \quad \mathbf{x}(0) = \mathbf{x_0}, \\ \mathbf{y}(t) &= h(\mathbf{x}(t)) \end{aligned} \tag{1}$$

with $t \in \mathbb{R}^+$, $\mathbf{x}(t) \in X \subseteq \mathbb{R}^n$, $\mathbf{u}(t) \in U \subseteq \mathbb{R}^m$, $\mathbf{y}(t) \in \mathbb{R}^m$, and f, h being smooth functions.

Given a reference trajectory $\mathbf{y}_{ref}(t)$, determine an optimal control problem (OCP) for real-time MPC that can be used to compute an input $\mathbf{u}(t)$ such that high-performance tracking is achieved, i.e. $||\mathbf{y}(t) - \mathbf{y}_{ref}(t)||$ remains small.

To achieve this, we assume that (1) is *differentially flat*, see Section III-A, and propose a Flatness-based Model Predictive Control (FMPC) that utilizes this property.

III. BACKGROUND

A. Differential Flatness

We recall the formal definition of differential flatness.

Definition 1: A nonlinear system model (1) is *differentially flat* if there exists $\boldsymbol{\zeta}(t) \in \mathbb{R}^m$, whose components are differentially independent (that is, the components are not related to each other through a differential equation), such that the following holds [13]:

$$\boldsymbol{\zeta} = \Lambda(\mathbf{x}, \mathbf{u}, \dot{\mathbf{u}}, \dots, \mathbf{u}^{(\delta)}), \tag{2}$$

$$\mathbf{x} = \Phi(\boldsymbol{\zeta}, \dot{\boldsymbol{\zeta}}, \dots, \boldsymbol{\zeta}^{(\rho-1)}), \tag{3}$$

$$\mathbf{u} = \Psi^{-1}(\boldsymbol{\zeta}, \dot{\boldsymbol{\zeta}}, \dots, \boldsymbol{\zeta}^{(\rho)}), \tag{4}$$

where Λ, Φ and Ψ^{-1} are smooth functions, δ and ρ are the maximum orders of the derivatives of $\mathbf{u}$ and $\boldsymbol{\zeta}$ needed to describe the system and $\boldsymbol{\zeta} = [\zeta_1, \dots, \zeta_m]^T$ is called the flat output.

B. Feedforward Linearization

We briefly highlight a key result in feedforward linearization that demonstrates how we can rewrite the nonlinear model (1) as an equivalent linear one. As explained in [15], every *differentially flat* system (1) can be represented using a Brunovský state (or *flat state*):

$$\mathbf{z} := \left[\zeta_1, \dot{\zeta}_1, \dots, \zeta_1^{(\rho_1-1)}, \dots, \zeta_m, \dots, \zeta_m^{(\rho_m-1)}\right]^T. \tag{5}$$

Note that ρ_i is the maximum derivative of ζ_i found in (4). Using the state transformation between the flat state $\mathbf{z}$ and state $\mathbf{x}$, obtained by differentiation of (2) and using (3), we can transform (1) into the normal form:

$$\zeta_i^{(\rho_i)} = \alpha_i(\boldsymbol{\zeta}, \dot{\boldsymbol{\zeta}}, \dots, \boldsymbol{\zeta}^{(\rho-1)}, \mathbf{u}, \dot{\mathbf{u}}, \dots, \mathbf{u}^{(\sigma_i)}) := v_i, \tag{6}$$

where α_i, $i = 1 \dots m$, is a smooth function obtained as a result of the transformation. Note σ_i is the maximum derivative of $\mathbf{u}$ after ρ_i times differentiating ζ_i in (2). We define the *flat input* $\mathbf{v}$ as:

$$\mathbf{v} := \left[v_1, v_2, \dots, v_m\right]^T. \tag{7}$$

Using the definitions in (5) and (7), we rewrite (6) as:

$$\dot{\mathbf{z}} = \mathbf{A}\mathbf{z} + \mathbf{B}\mathbf{v}, \tag{8a}$$

$$\mathbf{v} = \Psi(\mathbf{z}, \mathbf{u}, \dot{\mathbf{u}}, \dots, \mathbf{u}^{(\sigma)}), \tag{8b}$$

where $\sigma = \max \sigma_i$. We term (8a) the *linear flat model.* By substituting the definitions in (5) and (7), we can rewrite (4) as $\mathbf{u} = \Psi^{-1}(\mathbf{z}, \mathbf{v})$.

Theorem 1: (obtained from [15]) Consider a desired trajectory in the flat output $\boldsymbol{\zeta}_d$, including a corresponding desired flat state $\mathbf{z}_d$ (obtained by substituting $\boldsymbol{\zeta}_d$ for $\boldsymbol{\zeta}$ in (5)) and desired flat input $\mathbf{v}_d$ (obtained by substituting $\boldsymbol{\zeta}_d$ for $\boldsymbol{\zeta}$ in (6) and (7)). Given $\boldsymbol{\zeta}_d$, if we apply the nominal control,

$$\mathbf{u} = \Psi^{-1}(\mathbf{z}_d, \mathbf{v}_d), \tag{9}$$

to a differentially flat system (1), provided that $\mathbf{z}(0) = \mathbf{z}_d(0)$, this results in an equivalent, by change of coordinates, linear system as given in (8a).

Theorem 1 allows trajectory generators or controllers, as in our proposed approach in Fig. 1, to only consider the equivalent linear flat model. The output of the trajectory generator or controller, i.e., the desired flat state and flat input, can then be fed through the inverse transformation (9) to correct for the nonlinear part (8b) in the system. Feedforward linearization differs fundamentally from feedback linearization in that the desired flat state, see (9), as opposed to the measured flat state is used in the inverse term.

IV. RELATED WORK

Our proposed approach founds itself on an intuitive idea presented in early work using MPC and feedback linearization (MPC+FBL). In [10], the coupling of MPC and inner loop feedback linearization, $\mathbf{u} = \Psi^{-1}(\mathbf{z}, \mathbf{v}_d)$, is proposed. The idea is simply to cancel the nonlinear terms in (8b) and be left with linear prediction dynamics (8a), where $\mathbf{v} = \mathbf{v}_d$. The resulting MPC considers (8a) instead of the nonlinear model (1) with output $\mathbf{y}(t) = \boldsymbol{\zeta}(t)$. Despite this computational advantage, there are a few factors that must be considered:

Factor 1 – Development of a new cost: The required convexity of the resulting optimal control problem may force us to develop a new cost function. Consider that even if the original cost function used for the nonlinear system is convex, the nonlinear transformation of the state $\mathbf{x}$ and input $\mathbf{u}$ into flat state $\mathbf{z}$ and flat input $\mathbf{v}$ can result in a nonlinear, non-convex cost for the new flat variables [10].

Factor 2 – Conversion of input constraints: Another critical consideration is that due to the nonlinear parametrization in (4) convex constraints on the system input $\mathbf{u}$ may not map to convex constraints on the flat input $\mathbf{v}$. In general, there are two approaches in the literature to obtain linear input constraints on $\mathbf{v}$. The first approach calculates the exact input constraint on $\mathbf{v}$ at the current time step and then applies this as a constant constraint for the entire prediction horizon [17]. The second approach uses the previously predicted solution sequence for the flat state to construct a linear approximation of the constraints on $\mathbf{v}$ [18], [19]. Our proposed FMPC also suffers from this limitation. In practice, we apply conservative box constraints on the quadrotor flat states and flat inputs as is done in [20] for trajectory generation. Determining less conservative constraints in a comparably efficient way is left for future work.

Factor 3 – Robustness: In [12], the authors conclude that coupling linear MPC and feedback linearization relies on cancellation of nonlinear terms, which makes its robustness to noise, parameter uncertainty and disturbances difficult to quantify. They suggest trying to incorporate plant uncertainty into MPC+FBL. Another robustness issue that has not been addressed for MPC+FBL is how to account for known input time delays. Our proposed FMPC, similar to [21], instead combines linear MPC with feedforward linearization. However, we extend their initial results beyond single-input single-output (SISO) simulations.

V. METHODOLOGY

Our proposed FMPC in Fig. 1 still requires careful consideration of *Factor 1* and *Factor 2* of MPC+FBL. However, it attempts to address some of the issues related to *Factor 3*. We select output $\mathbf{y}(t)$ to be the flat output $\boldsymbol{\zeta}(t)$. Similarly, the reference trajectory $\mathbf{y}_{ref}(t)$ is defined in the flat output space, i.e., $\boldsymbol{\zeta}_{ref}(t)$. The implementation steps of our proposed FMPC are:

Feedforward Linearization: The proposed coupling of feedforward linearization and MPC, as seen in Fig. 1, allows us to use the linear flat model (8a) in a feedback MPC. The MPC outputs $\mathbf{z}_d$ and $\mathbf{v}_d$, which are then fed through the inverse term (9). We take advantage of the robustness of feedforward linearization to parameter uncertainties, where unlike feedback linearization the inverse term does not try to explicitly cancel nonlinear terms. Further, unlike feedback linearization, we can only use feedforward linearization to reduce the nonlinear model (1) to an equivalent linear flat model (8a) because we satisfy the initial condition requirement in Theorem 1. We continuously ensure adherence to the initial condition requirement by feeding back our measured flat state $\mathbf{z}$ into the MPC where we re-optimize for our updated desired trajectory, $\mathbf{z}_d$ and $\mathbf{v}_d$. This means that feedback MPC and feedforward linearization have a symbiotic relationship: feedforward linearization allows us to use a simplified model in MPC, while using MPC as our feedback controller allows us to satisfy the conditions for feedforward linearization.

Model Predictive Control: A standard *direct* method MPC strategy is considered. At each sampling time, we solve an open-loop OCP by minimizing a convex quadratic cost function $J(\cdot)$, which is dependent on the sequence of predicted flat states $\hat{\mathbf{z}}$ and flat inputs $\hat{\mathbf{v}}$. This is subject to both the discretized linear flat model of the system (8a) and linear constraints on the flat state and input, which approximate state $\mathbf{x}(t) \in X$ and input $\mathbf{u}(t) \in U$ constraints. The resulting OCP is a convex QP which can be efficiently solved for a global minimum.

Time Delays: Our proposed FMPC, which couples MPC and feedforward linearization, can easily be extended to systems with known input time delays. Consider the nonlinear system model (1) but now the input has a known time delay t_d, i.e. $\dot{\mathbf{x}}(t) = f(\mathbf{x}(t), \mathbf{u}(t - t_d))$. In our differential flatness definition, (4) now becomes $\mathbf{u}(t - t_d) = \Psi^{-1}(\boldsymbol{\zeta}(t), \dot{\boldsymbol{\zeta}}(t), \ldots, \boldsymbol{\zeta}^{(\rho)}(t))$ or more compactly, using (5) and (7), $\mathbf{u}(t - t_d) = \Psi^{-1}(\mathbf{z}(t), \mathbf{v}(t))$. This gives an inverse term:

$$\mathbf{u}(t) = \Psi^{-1}(\mathbf{z}(t + t_d), \mathbf{v}(t + t_d)).$$

Notice that this relies on forward predicted states and so feedback linearization using the current flat state $\mathbf{z}(t)$ would not cancel the nonlinear terms. Our proposed FMPC instead feeds forward $\mathbf{z}_d(t + t_d)$ and $\mathbf{v}_d(t + t_d)$.

VI. SIMULATION EXAMPLE

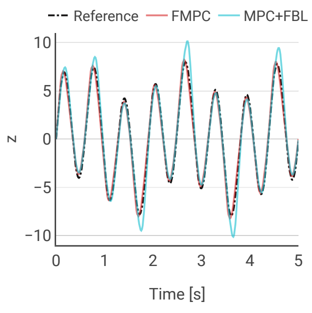
We compare our proposed FMPC, coupling MPC with feedforward linearization, with MPC+FBL in simulation for a SISO system. We consider a nonlinear system with the following nominal SISO model (taken from [15]):

$$\dot{x} = -x - x^3 + u, \tag{10}$$

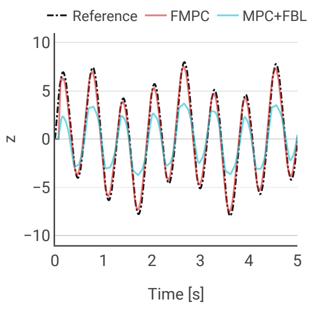
where, utilizing its differential flatness property, we can define flat state and flat output $z = \zeta = x$ and flat input $v = \dot{x}$. Equivalently, we rewrite the nonlinear model as a linear flat model $\dot{z} = v$ and a nonlinear term $v = -z - z^3 + u$. In the simulation, we consider the following MPC formulation:

$$\min_{\zeta_{1\ldots N}, v_{0\ldots N-1}} \frac{1}{2}\sum_{k=1}^{N} \tilde{Q}(\zeta_k - \zeta_{ref,k})^2 + \frac{1}{2}\sum_{k=0}^{N-1} \tilde{R} v_k^2,$$

subject to the discretized linear flat model, $z_{k+1} = z_k + \Delta t v_k$. We use a discretization of 70 Hz, prediction horizon $N = 70$ and weight matrices $\tilde{Q} = 100$ and $\tilde{R} = 0.1$. We consider a reference trajectory, in the flat output space, $\zeta_{ref}(t) = 2\sin(3t) + 6\sin(10t)$. The difference between our proposed FMPC and MPC+FBL is that the desired state (output from MPC) z_d, instead of the current flat state z, is used in the inverse term, $\Psi^{-1}(\cdot)$, i.e., $u = z_d + z_d^3 + v_d$ in FMPC versus $u = z + z^3 + v_d$ in MPC+FBL.



(a) *Model Parameter Sensitivity*



(b) *Input Time Delay*

Fig. 2. Tracking of reference $\zeta_{ref}(t) = 2\sin(3t) + 6\sin(10t)$ using FMPC and MPC+FBL for: (a) *Model Parameter Sensitivity*: with model parameter mismatch; (b) *Input Time Delay*: with known input time delay of 5 time steps.

We compare results for three cases:

Nominal Case: We consider a nonlinear system with the same dynamics as our nominal model (10). FMPC and MPC+FBL exhibit comparable performance with root mean square (RMS) error of 0.5547 and 0.3854, respectively. In this case, MPC+FBL is slightly better as exact cancellation of the nonlinearity is possible.

Model Parameter Sensitivity Case: To test robustness to parametric uncertainty, we use model (10) but consider a nonlinear system with dynamics $\dot{x} = -0.9x - 0.9x^3 + u$ (taken from [15]). FMPC, with RMS error 0.4826, has improved performance over MPC+FBL, with RMS error 1.0969. This is observed in Fig. 2(a) where inexact cancellation in MPC+FBL (light blue) results in the addition of unstable terms leading to poorer tracking performance.

Input Time Delay Case: We consider a nonlinear system with dynamics $\dot{x} = -x - x^3 + u(t - t_d)$ where t_d is a known time delay. We simulate the case $t_d = 5/70$. To compensate for this delay, our FMPC feeds forward $z_d(t + t_d)$ and $v_d(t + t_d)$ (computed by the MPC) as the desired flat state and flat input. In MPC+FBL, we similarly attempt to compensate by sending $v_d(t + t_d)$. FMPC achieves an RMS error of 0.6956 while MPC+FBL achieves 2.3150. Fig. 2 shows the success of such time delay compensation in FMPC (red), while the same approach cannot be used for MPC+FBL (light blue). Alternative approaches for time delay compensation in MPC+FBL may require an additional state predictor.

VII. EXPERIMENTS

A. Application to Quadrotors

We consider a cascaded control structure with a low-level onboard controller and an MPC outer-loop controller that can send commands $(\dot{z}_{cmd}, \phi_{cmd}, \theta_{cmd}, \dot{\psi}_{cmd})$, where $\dot{z}_{cmd}$ is commanded velocity in the z direction, ϕ_{cmd} is the commanded roll angle, θ_{cmd} is the commanded pitch angle and $\dot{\psi}_{cmd}$ is the commanded yaw rate. We compare LMPC, NMPC, MPC+FBL and FMPC.

To enhance trajectory tracking, we first perform a simple system identification, as in [6], to approximate the inner-loop attitude dynamics by:

$$\dot{\phi} = \frac{1}{\tau_\phi}(k_\phi \phi_{cmd} - \phi), \tag{11a}$$

$$\dot{\theta} = \frac{1}{\tau_\theta}(k_\theta \theta_{cmd} - \theta), \tag{11b}$$

$$\dot{\psi} = \dot{\psi}_{cmd}, \tag{11c}$$

where τ_ϕ, τ_θ are identified time constants, k_ϕ, k_θ are identified gains and ϕ, θ, ψ are the roll, pitch and yaw angles of the vehicle. Unlike in [6], we do not directly send a thrust command T_{cmd}. Consequently, we perform a similar system identification to approximate the z-velocity dynamics by a second-order response with a time delay of $t_d = 0.1s$:

$$\dddot{z}(t) = -\frac{1}{\tau_z}\dot{z}(t) - \frac{1}{\tau_{Iz}}\ddot{z}(t) + \frac{1}{\tau_{Cz}}\dot{z}_{cmd}(t - t_d), \tag{12}$$

where $\tau_z, \tau_{Iz}, \tau_{Cz}$ are identified time constants. Ignoring drag and other external forces, we can describe the lateral motion using the standard model [15], [5]:

$$\ddot{x} = \frac{\mathbf{R}_{13}}{\mathbf{R}_{33}}(\ddot{z} + g), \tag{13a}$$

$$\ddot{y} = \frac{\mathbf{R}_{23}}{\mathbf{R}_{33}}(\ddot{z} + g), \tag{13b}$$

where x, y, z represent the linear position, $\mathbf{R}$ the rotation of the quadrotor body frame with respect to an inertial frame and g the gravitational constant. We use the notation $\mathbf{R}_{13}$ to refer to the $(1, 3)$ entry of $\mathbf{R}$.

Nonlinear Model: In our NMPC model formulation, we consider the nonlinear model $\dot{\mathbf{x}} = f(\mathbf{x}, \mathbf{u})$ described by (11a)-(11c), (12) and (13a)-(13b) with state and input:

$$\mathbf{x} = (x, y, z, \dot{x}, \dot{y}, \dot{z}, \ddot{z}, \phi, \theta, \psi),$$
$$\mathbf{u} = (\dot{z}_{cmd}, \phi_{cmd}, \theta_{cmd}, \dot{\psi}_{cmd}).$$

Linearized Model: The only nonlinearity in our model formulation for NMPC comes from (13a)-(13b). In LMPC we consider the linearization of (13a)-(13b) about hover ($\phi = 0$, $\theta = 0$, $\ddot{z} = 0$) where at each time step we assume that our current yaw angle remains constant.

Linear Flat Model: The differential flatness of the standard quadrotor model for flat outputs $\boldsymbol{\zeta} = (x, y, z, \psi)$ is found in [14]. In a similar procedure, we can show the differential flatness, with the same flat outputs, of our nonlinear model governed by (11a)-(11c), (12) and (13a)-(13b). Our FMPC model formulation considers the linear flat model (8a) with flat state and flat input:

$$\mathbf{z} = (x, \dot{x}, \ddot{x}, y, \dot{y}, \ddot{y}, z, \dot{z}, \ddot{z}, \psi), \quad \mathbf{v} = (\dddot{x}, \dddot{y}, \dddot{z}, \dot{\psi}).$$

Optimal Control Problem: In the cost function in (14), $\boldsymbol{\zeta}_k = (x, y, z, \psi)_k$ is our flat output at time step k, the positive semi-definite matrix $\tilde{\mathbf{Q}} \succeq 0$ weights the error with our reference trajectory and the positive-definite $\tilde{\mathbf{R}} \succ 0$ regulates both the size and change in inputs $\mathbf{u} = (\dot{z}_{cmd}, \phi_{cmd}, \theta_{cmd}, \dot{\psi}_{cmd})$. In the LMPC OCP in (14), we optimize for $\mathbf{u}_k$ subject to the discretized *Linearized Model*. In NMPC, we similarly optimize for $\mathbf{u}_k$ using the cost in (14) but instead subject to the discretized *Nonlinear Model*.

In LMPC, we use a *direct* method to set up an OCP that is repeatedly solved at each time step:

$$\min_{\mathbf{u}_{0...N-1}} \frac{1}{2}\sum_{k=1}^{N}(\boldsymbol{\zeta}_k - \boldsymbol{\zeta}_{ref,k})^T\tilde{\mathbf{Q}}(\boldsymbol{\zeta}_k - \boldsymbol{\zeta}_{ref,k}) + \frac{1}{2}\sum_{k=0}^{N-1}\mathbf{u}_k^T\tilde{\mathbf{R}}\mathbf{u}_k$$
$$\text{subject to} \quad \mathbf{x}_{k+1} = \mathbf{A}\mathbf{x}_k + \mathbf{B}\mathbf{u}_k,\ \boldsymbol{\zeta}_k = \mathbf{C}\mathbf{x}_k, \quad \mathbf{x}_k \in \Omega_x. \tag{14}$$

For both LMPC and NMPC, we consider the constraint set in (14) to be:

$$\Omega_x = \{\mathbf{x} \in \mathbb{R}^{10} \mid |\ddot{z}| < 0.5; |\theta| \leq 0.4; |\phi| \leq 0.4; \psi \in [0, \pi]\}.$$

In the FMPC OCP, our cost is subject to the *Linear Flat Model*. To apply FMPC with a similar quadratic cost that is convex in $\mathbf{v}_k$ (discrete flat inputs) we use the *Linearized Model* to obtain a linear relationship between input $\mathbf{u}$ and our flat state $\mathbf{z}$ and flat input $\mathbf{v}$: $\mathbf{u} = \mathbf{M}\mathbf{z} + \mathbf{N}\mathbf{v}$. We then use this linear relationship to obtain a similar representative convex cost function for FMPC by replacing $\mathbf{u}$ with its linear relationship in $\mathbf{z}$ and $\mathbf{v}$. Consequently, in FMPC we solve the following OCP at each time-step:

$$\min_{\mathbf{v}_{0...N-1}} \frac{1}{2}\sum_{k=1}^{N}(\boldsymbol{\zeta}_k - \boldsymbol{\zeta}_{ref,k})^T\tilde{\mathbf{Q}}(\boldsymbol{\zeta}_k - \boldsymbol{\zeta}_{ref,k}) + \frac{1}{2}\sum_{k=0}^{N-1}\mathbf{u}_k^T\tilde{\mathbf{R}}\mathbf{u}_k$$
$$\text{subject to} \quad \mathbf{z}_{k+1} = \mathbf{A}\mathbf{z}_k + \mathbf{B}\mathbf{v}_k,\ \boldsymbol{\zeta}_k = \mathbf{C}\mathbf{z}_k, \quad \mathbf{u}_k = \mathbf{M}\mathbf{z}_k + \mathbf{N}\mathbf{v}_k, \quad \mathbf{z}_k \in \Omega_z, \tag{15}$$

where, similar to [20], we approximate the state constraint set Ω_x with the constraint set:

$$\Omega_z = \{\mathbf{z} \in \mathbb{R}^{10} \mid |\ddot{z}| < 0.5; |\ddot{x}| \leq 7; |\ddot{y}| \leq 7; \psi \in [0, \pi]\}.$$

Time Delay Compensation: To compensate for the time delay in the z-direction in (12), both LMPC and NMPC output $\dot{z}_{cmd}(t+t_d)$. As described in Section V, FMPC feeds forward $z_d(t+t_d), \dot{z}_d(t+t_d), \ddot{z}_d(t+t_d), \dddot{z}_d(t+t_d)$ through the inverse term (9).

B. Experimental Setup

The experiments are conducted on a Parrot AR.Drone quadrotor with an overhead motion capture system estimating the state of the quadrotor. We interface with the quadrotor using the open-source Robot Operating System (ROS). Outer-loop MPC approaches are run off-board on a ThinkPad P50 with Intel Core i7-6700HQ Processor. We use the formulations described in Section VII-A, to compare five different off-board outer-loop model predictive controllers running at 70 Hz, namely: nonlinear model predictive control using a first-order Euler discretization for forward simulation (NMPC E1); nonlinear model predictive control using a fourth-order explicit Runge-Kutta discretization for forward simulation (NMPC RK4); linear model predictive control (LMPC); model predictive control and feedback linearization (MPC+FBL); and our proposed flatness-based model predictive control (FMPC). All controllers consider a prediction at 10 Hz and a look-ahead time of 1 s, where the prediction horizon is $N = 10$ in (14) and (15). For NMPC, a single iteration of an SQP, with Gauss-Newton Hessian approximation, is performed at each iteration. It is initialized using *warm-starting*. Furthermore, all optimization problems are solved using a *single-shooting* method whereupon the resulting QP is solved using CVXOPT in Python. For each controller we perform three trials of three different trajectories: *Trajectory 1:* The quadrotor follows a circular reference with radius 1 m and angular frequency 0.4π rad/s in the x-y plane. There is no yawing and the vehicle remains at a constant altitude. *Trajectory 2:* The quadrotor performs a 2 s step in x and z. There is no motion in the y-direction and no yawing. *Trajectory 3:* The quadrotor follows the circular reference from trajectory 1, but now also simultaneously yaws to $\pi/2$ while performing a step in z. We do this for both a good parameter estimation ($\tau_\phi = \tau_\theta = 0.25, k_\phi = k_\theta = 1.4$) and a poor parameter estimation ($\tau_\phi = \tau_\theta = 0.5, k_\phi = k_\theta = 1.0$) of the time constants and gains in the inner-loop dynamics model in (11a)-(11b).

C. Results

While all model predictive controllers solve one QP at each time step, as seen in Fig. 3, our proposed FMPC (red) and MPC+FBL (light blue) have an average computation between that for LMPC (purple) and NMPC (grey and green). The additional computation used in our proposed FMPC over LMPC leads to reduced RMS error for *Trajectory 2* and *Trajectory 3* in Fig. 3. This reduced error is attributed to FMPC accounting for the nonlinear effects of yawing and vertical acceleration on lateral motion in (13a)-(13b). In *Trajectory 1* this effect is negligible since the trajectory requires no yawing or vertical motion. Comparable performance is observed for LMPC and our proposed FMPC.

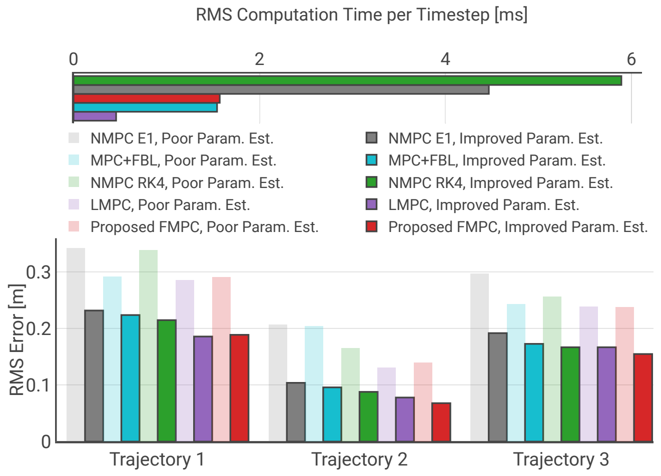
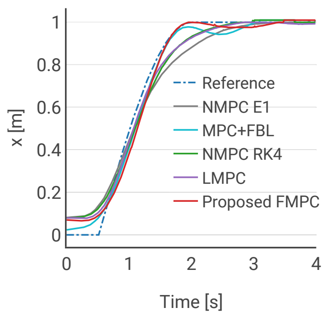
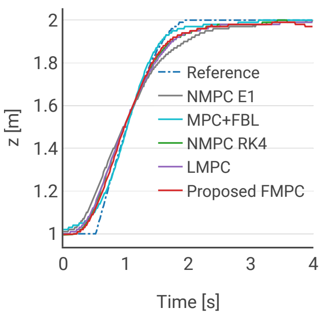


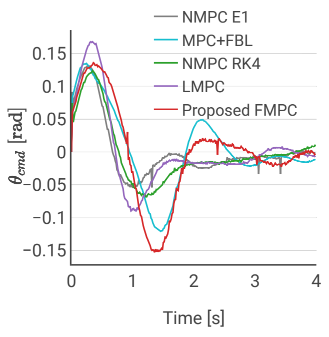
Fig. 3. Comparison of RMS computation per time step and RMS error for *Trajectory 1*, *Trajectory 2* and *Trajectory 3* (averaged over three trials per trajectory). Four key observations: (i) Our proposed FMPC and MPC+FBL require more computation than LMPC but less computation than NMPC (both NMPC E1 and NMPC RK4). (ii) However, for more aggressive trajectories (*Trajectory 2* and *Trajectory 3*), when good inner-loop parameter estimates are used, FMPC outperforms LMPC. (iii) Even with poor parameter estimates, FMPC shows comparable performance with LMPC. (iv) The relatively poor performance, for both good and poor parameter estimates, of NMPC RK4 and MPC+FBL is likely attributed to their sensitivity to nonlinear model accuracy.



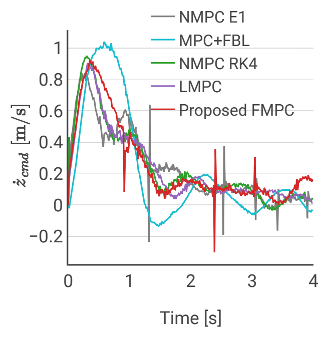
(a) Tracking in x



(b) Tracking in z



(c) Pitch command



(d) z-velocity command

Fig. 4. Comparison of tracking *Trajectory 2*: Improved performance in x-direction tracking by FMPC as a result of modified pitch command. Unlike LMPC, FMPC considers the effect of z-acceleration on lateral tracking. FMPC determines a larger pitch command for a longer period of time in the first 1 s allowing increased x acceleration before making a more substantial negative change than the other controllers.

As seen in Fig. 4, FMPC (red) accounts for the nonlinear effect of vertical acceleration on lateral motion in *Trajectory 2* by modifying the pitch command such that greater lateral acceleration is achieved in the first 1 s before allowing for a more dramatic change in pitch command to slow the vehicle down as it reaches $x = 1$ m. This tends to provide a tracking improvement of 5-15% over LMPC (purple). NMPC RK4 (green) achieves similar performance to LMPC potentially suggesting that for *Trajectory 2* linearization along the simulated trajectory (as is done for NMPC) provides little overall performance advantage over linearization about hover (as is done for LMPC). Interestingly, MPC+FBL tries to execute a similar tracking to our proposed FMPC but its performance is limited by its sensitivity to model inaccuracies and delays.

VIII. CONCLUSION

Results show that our proposed Flatness-based Model Predictive Control (FMPC) applied to quadrotor trajectory tracking is promising. Its advantages include:

- unlike LMPC, it is able to account for nonlinearities while still solving a convex QP;
- unlike NMPC, it is not sensitive to initial trajectory choice or susceptible to converging to local minima;
- using feedforward linearization instead of feedback linearization improves robustness to modelling errors and can account for known input time delays.

REFERENCES

[1] A. Bircher, M. Kamel, K. Alexis, H. Oleynikova, and R. Siegwart, "Receding horizon path planning for 3D exploration and surface inspection," *Autonomous Robots*, pp. 1-16, 2016.

[2] P. Rudol and P. Doherty, "Human body detection and geolocalization for UAV search and rescue missions using color and thermal imagery," in *Proc. IEEE Aerospace Conference*, 2008, pp. 1-8.

[3] J. Han and Y. Chen, "Multiple UAV formations for cooperative source seeking and contour mapping of a radiative signal field," *Journal of Intelligent & Robotic Systems*, pp. 1-10, 2013.

[4] M. Bangura and R. Mahony, "Real-time model predictive control for quadrotors," *IFAC Proceedings Volumes*, vol. 47, no. 3, pp 11773-11780, 2014.

[5] K. Alexis, C. Papachristos, R. Siegwart, and A. Tzes, "Robust model predictive flight control of unmanned rotorcrafts," *Journal of Intelligent & Robotic Systems*, pp. 1-27, 2015.

[6] M. Kamel, M. Burri, and R. Siegwart, "Linear vs nonlinear MPC for trajectory tracking applied to rotary wing micro aerial vehicles," *IFAC-PapersOnLine*, vol. 50, no. 1, pp. 3463-3469, 2017.

[7] Y. Wang and S.P. Boyd, "Fast model predictive control using online optimization," *IEEE Transactions on Control Systems Technology*, vol. 18, no. 2, pp. 267-278, 2010.

[8] M. Diehl , H.J. Ferreau, and N. Haverbeke (2009) Efficient numerical methods for nonlinear MPC and moving horizon estimation. In: Magni L., Raimondo D.M., Allgwer F. (eds) Nonlinear Model Predictive Control. Lecture Notes in Control and Information Sciences, vol 384. Springer, Berlin, Heidelberg

[9] B. Houska, H. Ferreau, and M. Diehl, "ACADO toolkit - An open source framework for automatic control and dynamic optimization," *Optimal Control Applications and Methods*, vol. 32, no. 3, pp. 298-312, 2011.

[10] J.A. Primbs and V. Nevistic, "MPC extensions to feedback linearizable systems," in *Proc. American Control Conference (ACC)*, 1997, pp. 2073-2077.

[11] V. Nevistic and M. Morari, "Robustness of MPC-based schemes for constrained control of nonlinear systems," *IFAC Proceedings Volumes*, vol. 29, no. 1, pp. 5823-5828, 1997.

[12] M.V. Khotare, V. Nevistic, and M. Morari, "Robust constrained model predictive control for nonlinear systems: a comparative study," in *Proc. IEEE Conference on Decision and Control (CDC)*, 1995, pp. 2884-2889.

[13] M. Fliess, J. Lévine, P. Martin, and P. Rouchon, "Flatness and defect of non-linear systems: introductory theory and examples," *International Journal of Control*, vol. 61, no. 6, pp. 1327-1361, 1995.

[14] D. Mellinger and V. Kumar, "Minimum snap trajectory generation and control for quadrotors," in *Proc. IEEE International Conference on Robotics and Automation (ICRA)*, 2011, pp. 2520-2525.

[15] V. Hagenmeyer and E. Delaleau, "Exact feedforward linearization based on differential flatness," *International Journal of Control*, vol. 76, no. 6, pp. 537-556, 2003.

[16] V. Hagenmeyer, S. Streif, and M. Zeitz, "Flatness-based feedforward and feedback linearisation of the ball&plate lab experiment," in *Proc. 6th IFAC-Symposium on Nonlinear Control Systems (NOLCOS)*, 2004, pp. 233-238.

[17] J. Deng, V. M. Becarra, and R. Stobart, "Input constraints handling in an MPC/feedback linearization scheme," *International Journal of Applied Mathematics and Computer Science*, vol. 19, no. 2, pp. 219-232, 2009.

[18] K. Margellos and J. Lygeros, "A simulation based MPC technique for feedback linearizable systems with input constraints," in *Proc. 49th IEEE Conference on Decision and Control (CDC)*, 2010, pp. 7539-7544

[19] M.J. Kurtz and M.A. Henson, "Feedback linearizing control of discrete-time nonlinear systems with input constraints," *International Journal of Control*, vol. 70, no. 4, pp. 603-616, 2010.

[20] M.W. Mueller and R. D'Andrea, "A model predictive controller for quadrocopter state interception," in *Proc. European Control Conference (ECC)*, 2013, pp. 1383-1389.

[21] C. Kandler, S.X. Ding, T. Koenings, N. Weinhold, and M. Schultalbers, "A differential flatness based model predictive control approach," in *Proc. IEEE International Conference on Control Applications (CCA)*, 2012, pp. 1411-1416.