A constrained instantaneous learning approach for aerial package delivery robots : onboard implementation and experimental results

Mehndiratta, Mohit; Kayacan, Erdal

2019

Mehndiratta, M., & Kayacan, E. (2019). A constrained instantaneous learning approach for aerial package delivery robots : onboard implementation and experimental results. *Autonomous Robots*, 43(8), 2209-2228. doi:10.1007/s10514-019-09875-y

<https://hdl.handle.net/10356/141838>

<https://doi.org/10.1007/s10514-019-09875-y>

© 2019 Springer Science+Business Media, LLC, part of Springer Nature. This is a post-peer-review, pre-copyedit version of an article published in *Autonomous Robots*. The final authenticated version is available online at:

<http://dx.doi.org/10.1007/s10514-019-09875-y>

Downloaded on 26 Apr 2023 19:13:26 SGT

A Constrained Instantaneous Learning Approach for Aerial Package Delivery Robots: Onboard Implementation and Experimental Results

Mohit Mehndiratta · Erdal Kayacan

Received: date / Accepted: date

Abstract Rather than utilizing a sophisticated robot which is trained – and tuned – for a scenario in a specific environment perfectly, most people are interested in seeing robots operating in various conditions where they have never been trained before. In accordance with the goal of utilizing aerial robots for daily operations in real application scenarios, an aerial robot must learn from its own experience and its interactions with the environment. This paper presents an instantaneous learning-based control approach for the precise trajectory tracking of a 3D-printed aerial robot which can adapt itself to the changing working conditions. Considering the fact that model-based controllers suffer from lack of modeling, parameter variations and disturbances in their working environment, we observe that the presented learning-based control method has a compelling ability to significantly reduce the tracking error under aforementioned uncertainties throughout the operation. Three case scenarios are considered: payload mass variations on an aerial robot for a package delivery problem, ground effect when the aerial robot is hovering/flying close to the ground, and wind-gust disturbances encountered in the outdoor environment. In each case study, parameter variations are learned using nonlinear moving horizon estimation (NMHE) method, and the estimated parameters are fed to the nonlinear model predictive controller (NMPC). Thanks to learning capability of the presented framework, the aerial robot can learn from its own experience, and react promptly – unlike iterative learning con-

trol which allows the system to improve tracking accuracy from repetition to repetition – to reduce the tracking error. Additionally, the fast C++ execution of NMPC and NMHE codes facilitates a complete onboard implementation of the proposed framework on a low-cost embedded processor.

Keywords Instantaneous learning, learning-based NMPC, NMPC-NMHE framework, unmanned aerial vehicle, tilt-rotor tricopter, package delivery, ground effect, wind-gust disturbance.

1 Introduction

Recent advances in sensor, actuation, and processing technologies as well as the availability of numerous low-cost onboard processors, e.g. ARM-based processors, have facilitated an unprecedented growth of multicopter unmanned aerial vehicles (UAVs) in various industries. They have been explored for various applications including, air package delivery (Amazon, 2017), search and rescue at disaster sites (Qi et al, 2016), damage evaluation after an earthquake (Chen et al, 2016), exploration tasks in an unknown environment (Bircher et al, 2016), and aggressive flights involved during perching (Mellinger et al, 2012). Researchers have also been working on developing algorithms and interfaces to support safe physical interactions and efficient user collaboration with UAVs in (Rajappa et al, 2017) and (Szafir et al, 2017), respectively.

Although their closed-loop control involves highly nonlinear and cross-coupled dynamics, many model-based controllers, such as proportional-integral-derivative (PID), linear-quadratic-regulator (LQR) (Bouabdallah et al, 2004) and model predictive control (MPC) (Bemporad et al, 2009), have been utilized successfully. However, these model-based controllers are perfectly tuned to operate in a particular scenario within an engineered (well-known) environment. This

Mohit Mehndiratta
School of Mechanical and Aerospace Engineering, Nanyang Technological University, 50 Nanyang Avenue, Singapore 639798, Singapore. Singapore Center for 3D Printing (SC3DP), Nanyang Technological University, 50 Nanyang Avenue, Singapore 639798, Singapore.
E-mail: mohit005@e.ntu.edu.sg

Erdal Kayacan
Department of Engineering, Aarhus University, DK-8000 Aarhus C.
E-mail: erdal@eng.au.dk

implies, there is no performance guarantee when they operate in an unknown environment, full of internal and external uncertainties. For instance, vision-based localization while navigating through unstructured environment induces additional uncertainties, which further degrades the performance of the perfectly tuned controllers. As a result, people in the robotics community are getting inclined towards controllers that can perform sufficiently well when exposed to unforeseen disturbances within an unknown environment. Therefore, in this paper, we analyze the control performance of an aerial robot in an environment which is substantially different than the environment controller is tuned for, by explicitly inducing operational disturbances in the form of mass variation, ground effect, and wind-gust.

Over other model-based control techniques, MPC owes its remarkable success to its unique ability to simultaneously handle constraints and optimize performance in a systematic and elegant manner via repetitive online optimization (Eren et al, 2017). Researchers are utilizing it for the control of various ground robots as demonstrated in (Kayacan et al, 2014a, 2015a, 2016), and for numerous multicopter aerial robots, including: (i) quadrotors for aerial manipulation tasks in (Garimella and Kobilarov, 2015; Seo et al, 2017), (ii) tilt-rotor tricopters in (Alexis et al, 2016; Prach and Kayacan, 2017; Lee et al, 2018), and (iii) for fault-tolerant control of coaxial tricopters in (Mehndiratta and Kayacan, 2018b). In addition, it has also been employed for the cooperative localization control of multiple UAVs in (Dentler et al, 2018), wherein the sensor limitations are described with the help of potential functions. Apart from all the benefits that MPC brings to the control of robotic systems, there is a weak side associated with its performance. That is, being a model-based controller, its performance is only guaranteed for an accurate mathematical model of the system to be controlled. This implies that a precise modeling of the system is required in order to achieve an optimal performance from MPC, which is very tedious to perform for robotic systems (Mellinger et al, 2011; Kayacan et al, 2014b). Besides linearization related errors associated with utilizing a linear model for a nonlinear plant as obtained in (Kayacan et al, 2015b), the changing working conditions encountered outdoors, induce other types of system uncertainties that eventually degrade the performance of the model-based controllers. One of the artifices is to incorporate some learning mechanism that enables these originally static controllers to learn from their own experience and hence, adapt themselves to the changing environment.

In this paper, we propose a learning-based nonlinear MPC (NMPC) for the precise trajectory tracking control of an aerial robot, operating in a disturbed environment. While NMPC is responsible for the position tracking of the UAV, a nonlinear moving horizon estimator (NMHE) is employed to learn UAV model which is time-varying due to distur-

bances. The enticing feature of this control framework is the learning part that makes it adaptive to the changing working conditions. In order to mimic the situation of an (unknown) outdoor testing environment, the tracking performance of the learning-based NMPC is investigated – and simultaneously compared with conventional NMPC – for three test case scenarios, incorporating different types of disturbances: (i) sudden mass variations encountered in package delivery applications, (ii) ground effect by flying in close proximity to the ground, and (iii) wind-gust via two industrial fans. In all the aforementioned test cases, a significant change in the UAV model is expected and thus, the use of learning-based NMPC is justified for these applications. This paper is a major extension of the work presented in (Mehndiratta et al, 2019), wherein the NMHE is extended to estimate two other types of disturbances namely, ground effect and wind-gust, followed by a complete onboard implementation of the closed-loop framework on a low-cost embedded processor. In addition, some statistical results are also obtained in this paper that illustrate the viability of the closed-loop framework for increasing disturbance levels. To summarize, the key contributions of this paper are as follows:

1. Design of a learning-based NMPC to realize the precise position tracking of a custom-designed, fully 3D-printed (except electronics) tilt-rotor tricopter UAV.
2. Instantaneous learning of time-varying model parameters via NMHE.
3. Fast C++ NMPC and NMHE codes have facilitated an onboard implementation on a low-cost embedded processor, i.e., Raspberry Pi 3. To the best of our knowledge this is the first (simultaneous) implementation of these methods on such a low-cost processor.

This paper is organized as follows: Section 2 describes the other ongoing research in learning-based control field and relates our work to them. Section 3 presents the details of the utilized aerial robotic platform along with its mathematical model. In Section 4, control and estimation techniques, in terms of NMPC and NMHE problem formulations are presented. Section 5 illustrates the design of each component of the utilized closed-loop framework. Thereafter, the experimental results are presented in Section 6. Lastly, in Section 7, some conclusions are drawn from this paper.

2 Related Work

Various types of learning-based control schemes exist in the literature. Amongst them, the most prominent ones are adaptive control, iterative learning control (ILC), and instantaneous learning control. Each of these schemes, along with the related recent research, are discussed in the subsequent paragraphs.

The adaptive controllers modify the controller parameters in order to match the performance of a reference model. A well-utilized approach is the adaptive-optimal control strategy which comprises of the use of an adaptive controller for stability during the learning phase, followed by switching to the main model-based optimal controller that eventually optimizes the performance (Chowdhary et al, 2013a,b). An online switching metric is developed that initiates the switching to MPC after gaining enough confidence in the parameter estimates. Since the adaptive controller aims to match the performance of a reference model, it requires an accurate identification of the system in the first place. This is in contrast with our learning-based NMPC approach, wherein we learn the model parameters during operation, and hence, a precise *a priori* model is not mandatory. Other types of control approaches that facilitate the adaptation of the system to a certain extent are: gain scheduling control (Ataka et al, 2013) and control based on switching between multiple models (Alexis et al, 2011). While computing the control gains for various flight conditions is quite tedious, in-flight switching between multiple models may destabilize the closed-loop. Recently, a simple learning strategy has been demonstrated to improve the tracking performance of traditional feedback linearization controller for an aerial robot in (Mehndiratta et al, 2018). However, feedback linearization control has its own limitations over MPC, which includes the plausible instability of the zero dynamics in the presence of model uncertainties.

In various model learning control frameworks, the basic idea is to learn the uncertain (and/or time-varying) model parameters throughout the operation. These learned parameters are then modified within the controller, in contrast to the controller parameters as done in adaptive control. Two main types of model learning approaches exist: iterative learning and instantaneous learning. The ILC technique which is based on the iterative learning paradigm utilizes experience from the system repetitions in order to identify the uncertain dynamics. In many applications, the unknown dynamics is modeled as Gaussian process (GP) because it effectively captures the complex nonlinear behaviors. For instance, (Ostafew et al, 2016) utilized this ILC scheme in order to improve the tracking performance of NMPC for a ground robot which is operating in an uncertain terrain. The disturbances are modeled as GPs, which actually represent the discrepancies between the *a priori* model and the measured system behavior. In addition, (Cao et al, 2016) implemented the ILC technique for the control of an unmanned quadrotor. In their application, the whole quadrotor dynamics is modeled using GP and is learned from the beginning, purely utilizing observations. Another UAV related ILC application is presented in (Schoellig et al, 2012), where an optimization-based learning strategy is incorporated that updates the feed-forward input signal for a precise trajectory

tracking application of a quadrotor. In ILC scheme, however, a few trial runs are required before the uncertain model gets identified fully. This is not the case with our approach, as it is based on the instantaneous learning which enables the feed of the estimated model to the controller just after a few observations.

Iterative learning methods may work well in repetitive operations where the statistical properties of the uncertainty do not vary drastically; e.g., a ground robot application for agricultural operations where soil conditions do not vary remarkably from one repetition to another. On the other hand, there are some robotic applications, such as aerial robots where wind-gust disturbances play a critical role in the performance, while the statistical properties of the uncertainties vary significantly over time. Therefore, an instant adaptation approach is needed throughout the operation without waiting for the next repetition. The instantaneous (online) learning scheme utilizes estimation algorithms in order to identify the uncertain or time-varying model parameters, and subsequently, feed them to the controller. Numerous implementations have been made for the instantaneous learning-based scheme, incorporating various estimators. For instance, (Shin et al, 2010) utilized a support vector regression method to learn the lateral dynamical model of a UAV. In (Garcia and Keshmiri, 2013), an artificial neural network-based estimator is employed in order to identify the model of a meridian unmanned aerial system, operating in off-nominal conditions. Other applications of instantaneous learning-based scheme include: an onboard control of a quadrotor using modified extended Kalman filter (EKF) for learning in (Bouffard et al, 2012), control of a 3 degree of freedom helicopter setup utilizing NMHE (and EKF) - based learning of aerodynamic coefficients in (Mehndiratta and Kayacan, 2017), online learning-based cascade control of tilt-rotor tricopter UAV with varying thrust and drag-moment coefficients in (Mehndiratta and Kayacan, 2018a), and centralized control of autonomous trailer-tractor system utilizing NMHE-based learning of slip parameters in (Kayacan et al, 2015a). Although we are incorporating a similar learning strategy with the above applications, the factors that differentiate our work from the others are: (i) we utilize NMHE for estimation which can deal with constraints in its formulation and is better in performance in comparison to EKF, as illustrated from our previous work in (Mehndiratta and Kayacan, 2017), and (ii) we explicitly induce three types of operational uncertainties, and learn the varying model parameters in real-time for better tracking accuracy.

Fig. 1: SolidWorks CAD drawings of the 3D-printed tilt-rotor tricopter UAV.

Fig. 2: 3D-printed hub plates of the tilt-rotor tricopter UAV.

3 Prototyped Aerial Robot for Package Delivery With an Automated Dropping Mechanism

3.1 System Description

The aerial robot used in this application is a 3D-printed tilt-rotor tricopter UAV. Two views of its SolidWorks CAD drawing are shown in Fig 1. For the package delivery problem considered in this paper, it is a fully custom-designed UAV which is inspired by commercial Talon tricopter frames available in the market. The major design requirement was to obtain a small and light-weight platform in order to suit our constrained real-time testing environment. In addition, the required payload dropping mechanism and electronics are integrated into the frame design to provide a compact, stable and lightweight UAV system. Guide rails are printed on the UAV hub base to hold the heaviest component, the battery, in the center of the UAV, thereby reducing the mass center deviation and providing greater stability. Attachment points are also printed on the hub plates to hold the servomotor for the payload dropping mechanism. Additionally, grommet holders are embedded with the hub top to provide damping for the flight controller. All aforementioned features can be seen in the images of the 3D-printed base and top hub plates, depicted in Fig. 2.

The payload dropping mechanism, mounted at the base of the UAV, consists of two plates. Amongst them, one houses the servomotor, while the other holds the payload blocks to be dropped. These payload blocks are designed

with hooks, which are positioned within slots of the latter plate. For the dropping mechanism, a circular gear which is attached to the servomotor drives a linear gear that results in a translation motion of a push rod. This translation motion of the rod eventually opens the interlocking for blocks and thus, allows them to be dropped sequentially. The entire assembly of the dropping mechanism is shown in Fig. 3.

Pixhawk flight controller is incorporated to provide low-level stabilization. Additionally, the underside of the UAV houses an onboard computer (Raspberry Pi 3), which besides running the control and estimation codes, also controls the servomotor for the dropping mechanism. Finally, the two views of the whole assembled UAV in flight are presented in Fig. 4. As desired, it is a compact UAV with span and length of 42.4 cm and 37.6 cm, respectively. Its overall weight is around 1.018 kg including all electronics except the battery.

3.2 System Modeling

The considered tilt-rotor tricopter has two stationary rotors in the front – RR (right rotor) rotating clockwise and LR (left rotor) rotating counter-clockwise – and one non-stationary (or tilting) rotor at the back – BR (back rotor) rotating counter-clockwise – as shown in Fig. 5.

The translational kinematic equations, describing position of the UAV, are obtained using the transformation from body frame ($\mathcal{F}_B$) to Earth-fixed frame ($\mathcal{F}_E$)(Bouabdallah, 2007):

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \end{bmatrix} = R_{EB} \begin{bmatrix} u \\ v \\ w \end{bmatrix}, \quad (1)$$

where x, y, z represent the translational position which is defined in frame $\mathcal{F}_E$; u, v, w are the translational velocities that are defined in frame $\mathcal{F}_B$; R_{EB} is the translation transformation matrix between frames $\mathcal{F}_E$ and $\mathcal{F}_B$, and is expressed as ($c : \cos, s : \sin, t : \tan$):

$$R_{EB} = \begin{bmatrix} c\theta c\psi & s\phi s\theta c\psi - s\psi c\phi & c\phi s\theta c\psi + s\phi s\psi \\ c\theta s\psi & s\phi s\theta s\psi + c\psi c\phi & c\phi s\theta s\psi - s\phi c\psi \\ -s\theta & s\phi c\theta & c\phi c\theta \end{bmatrix}, \quad (2)$$

Fig. 3: Inverted views of the payload dropping mechanism.

(a) Side view.

(b) Isometric view.

Fig. 4: Prototyped 3D-printed tilt-rotor tricopter UAV.

where ϕ , θ , ψ represent rotational attitude of the UAV defined in frame $\mathcal{F}_E$. In addition, the rigid-body dynamic equations of a tilt-rotor tricopter are derived based on the

Fig. 5: Coordinate frame and sign conventions for the considered tilt-rotor tricopter UAV.

Notation: $[\Omega_1, \Omega_2, \Omega_3]$ - rotational RPM of three rotors; μ - tilting angle of back rotor; $[F_1, F_2, F_3]$ - forces generated by three rotors; $[\tau_1, \tau_2, \tau_3]$ - moments generated by three rotors; $[\tau_x, \tau_y, \tau_z]$ - external moments acting on the tricopter body.

Newton-Euler formulation in the body coordinate system:

$$\dot{u} = rv - qw + g \sin(\theta) + \frac{1}{m} F_x, \quad (3a)$$

$$\dot{v} = pw - ru - g \sin(\phi) \cos(\theta) + \frac{1}{m} F_y, \quad (3b)$$

$$\dot{w} = qu - pv - g \cos(\phi) \cos(\theta) + \frac{1}{m} F_z, \quad (3c)$$

where F_x , F_y , F_z are the total external forces acting on the tricopter body in frame $\mathcal{F}_B$. In general, these external forces are generated when a rotor is propelled through the surrounding air and are modeled depending on the number of rotors in the system. Therefore, using the momentum theory, a hovering rotor can be modeled as: $F_i = K_F \Omega_i^2$, where F_i is the force generated by i^{th} rotor with Ω_i angular velocity and K_F represents the aerodynamic parameter of the rotor which is commonly referred as the force coefficient. In addition to the propulsive force, other perturbation forces including wind-gust, also act on the UAV during its outdoor operation. Therefore, the expression for the total external force acting on the tricopter body in $\mathcal{F}_B$ frame is written as:

$$F_{\text{ext}} = \begin{bmatrix} F_x \\ F_y \\ F_z \end{bmatrix} = \begin{bmatrix} F_{x\text{Dist}} \\ -F_3 \sin(\mu) + F_{y\text{Dist}} \\ F_1 + F_2 + F_3 \cos(\mu) + F_{z\text{Dist}} \end{bmatrix}, \quad (4)$$

where μ is the tilting angle of the back rotor, and $F_{x\text{Dist}}$, $F_{y\text{Dist}}$ and $F_{z\text{Dist}}$ are the considered disturbance forces along x -, y - and z -directions, respectively. In general, these disturbance forces cannot be measured directly, and hence, an observer is employed to estimate them.

The constant intrinsic parameters for the custom-designed 3D-printed tilt-rotor tricopter UAV are listed in Table 1. While parameters such as mass (m) and the arm lengths (l_1, l_2, l_3) are directly measured, the thrust coefficient is evaluated based on a simple experimentation. It involves measuring thrust generated by the propeller for various rotor RPMs and later plotting them together in order to compute the slope of the plot. Regarding the details of this experiment, one may refer (Li et al, 2018).

For the purpose of real-time implementation, the nonlinear tricopter model is discretized incorporating the direct multiple shooting method with a shooting grid size of $\Delta t_s = 0.01s$. In addition, the explicit Runge-Kutta 4th order integrator is utilized with 2 steps per shooting interval. Finally, the full-state discretized nonlinear model of the tilt-rotor tricopter UAV can be represented as:

$$\mathbf{x}_{k+1} = \mathbf{f}_d(\mathbf{x}_k, \mathbf{u}_k, \mathbf{p}), \quad (5)$$

$$\mathbf{z}_k = \mathbf{h}(\mathbf{x}_k, \mathbf{u}_k), \quad (6)$$

where the state vector $\mathbf{x} \in \mathbb{R}^{n_x}$ and control vector $\mathbf{u} \in \mathbb{R}^{n_u}$ are:

$$\mathbf{x} = [x, y, z, u, v, w, \phi, \theta, \psi, p, q, r]^T, \quad (7)$$

$$\mathbf{u} = [\Omega_1, \Omega_2, \Omega_3, \mu]^T, \quad (8)$$

Table 1: 3D-printed tilt-rotor tricopter intrinsic parameters, as visualized in Fig. 5.

Par.	Description	Value
m	Mass of tricopter UAV	1.442 kg
l_1	Moment arm	0.284 m
l_2	Moment arm	0.212 m
l_3	Moment arm	0.092 m
I_{xx}	MOI about $\mathcal{F}_{B_x}$	0.016053 kg-m ²
I_{yy}	MOI about $\mathcal{F}_{B_y}$	0.028158 kg-m ²
I_{zz}	MOI about $\mathcal{F}_{B_z}$	0.032752 kg-m ²
I_{xz}	MOI about $\mathcal{F}_{B_{xz}}$	0.029763 kg-m ²
J_p	MOI of propeller	6×10^{-5} kg-m ²
K_f	Force coefficient	3.76×10^{-5} N-s ²
K_τ	Drag-moment coefficient	2.56×10^{-6} Nm-s ²

and $\mathbf{z} \in \mathbb{R}^{n_z}$ is the measurement vector. The term $\mathbf{p} \in \mathbb{R}^{n_p}$ denotes the system parameter vector. Also, the state and measurement functions are denoted by $\mathbf{f}_d(\cdot, \cdot, \cdot): \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \times \mathbb{R}^{n_p} \rightarrow \mathbb{R}^{n_x}$ and $\mathbf{h}(\cdot, \cdot): \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \rightarrow \mathbb{R}^{n_z}$, respectively.

Remark 1 Since the presented algorithm is not utilized to perform the low-level (attitude) control for the UAV, the rotational kinematic and dynamic equations are omitted for brevity. However, the interested readers are referred to (Mehndiratta et al, 2019) for an elaborate model of the tilt-rotor tricopter UAV.

4 Instantaneous Learning-based Nonlinear Model Predictive Control

4.1 Nonlinear Model Predictive Control

(N)MPC is an advanced, dynamic optimization-based strategy for feedback control. Its overall performance is primarily reliant on the accuracy of the system model. Conceptually, it computes an optimum current control action by optimizing model's behavior over a finite window, often recalled as prediction horizon (N_c). The optimal forecast of the system behavior results from an open-loop online optimization, represented in the form of a constrained, finite horizon, optimal control problem (OCP). The solution of this open-loop OCP for the current system's state and at the current time instant results in an optimal sequence of control actions; the first term of which is then regarded as the optimal control action (Eren et al, 2017). Moreover, the finite prediction window recedes forward in time, which results in a *receding horizon control* technique.

For our trajectory tracking application similar to (Vukov et al, 2015), we formulate the parametric OCP for NMPC utilizing a least square function that penalizes the deviations of predicted state and control trajectories from their specified references, over the given prediction horizon window ($t_j \leq t \leq t_{j+N_c}$). Therefore, a discrete time optimization

problem is of the form:

$$\min_{\mathbf{x}_k, \mathbf{u}_k} \frac{1}{2} \left\{ \sum_{k=j}^{j+N_c-1} \left(\left\| \mathbf{x}_k - \mathbf{x}_k^{\text{ref}} \right\|_{W_x}^2 + \left\| \mathbf{u}_k - \mathbf{u}_k^{\text{ref}} \right\|_{W_u}^2 \right) + \left\| \mathbf{x}_{N_c} - \mathbf{x}_{N_c}^{\text{ref}} \right\|_{W_{N_c}}^2 \right\} \quad (9a)$$

$$\text{s.t. } \mathbf{x}_j = \hat{\mathbf{x}}_j, \quad (9b)$$

$$\mathbf{x}_{k+1} = \mathbf{f}_d(\mathbf{x}_k, \mathbf{u}_k, \mathbf{p}), \quad k = j, \dots, j+N_c-1, \quad (9c)$$

$$\mathbf{x}_{k,\min} \leq \mathbf{x}_k \leq \mathbf{x}_{k,\max}, \quad k = j, \dots, j+N_c, \quad (9d)$$

$$\mathbf{u}_{k,\min} \leq \mathbf{u}_k \leq \mathbf{u}_{k,\max}, \quad k = j, \dots, j+N_c-1, \quad (9e)$$

where $\mathbf{x}_k \in \mathbb{R}^{n_x}$ is the differential state, $\mathbf{u}_k \in \mathbb{R}^{n_u}$ is the control input and $\hat{\mathbf{x}}_j \in \mathbb{R}^{n_x}$ is the current state estimate; time-varying state and control references are denoted by $\mathbf{x}_k^{\text{ref}}$ and $\mathbf{u}_k^{\text{ref}}$, respectively; terminal state reference is represented by $\mathbf{x}_{N_c}^{\text{ref}}$; $W_x \in \mathbb{R}^{n_x \times n_x}$, $W_u \in \mathbb{R}^{n_u \times n_u}$ and $W_{N_c} \in \mathbb{R}^{n_x \times n_x}$ are the corresponding weight matrices, which are assumed constant for this application. However, their time-varying formulation can also be included in a similar manner. Furthermore, $\mathbf{x}_{k,\min} \leq \mathbf{x}_{k,\max} \in \mathbb{R}^{n_x}$ and $\mathbf{u}_{k,\min} \leq \mathbf{u}_{k,\max} \in \mathbb{R}^{n_u}$, specify the lower and upper bounds on the states and control inputs, respectively.

The last expression in (9a) represents the final cost incurred due to the finite prediction horizon and is generally referred as the terminal penalty term. This term is often included in the problem formulation for stability reasons (Kraus et al, 2013). In addition, some other stability results include a problem formulation with a sufficiently long horizon (Grüne, 2012), an additional prediction horizon, and a locally stabilizing control law (Nicolao et al, 1998). One may note that the stability and optimality proofs of NMPC are not explicitly included in this paper, however, the interested readers are referred to (Mayne et al, 2000; Diehl et al, 2001).

4.2 Nonlinear Moving Horizon Estimation

Typically, (N)MHE is considered as a *dual problem* of (N)MPC as they both exploit the same optimization problem structure; despite the fact that (N)MPC predicts the future of the system, while (N)MHE utilizes the past measurements over an estimation horizon for state estimation (Kühl et al, 2011). Moreover, the two main differences in the OCP formulation of (N)MHE and (N)MPC are: (i) there is no initial state constraint like (9b), and (ii) the optimization variables are the states and unknown system parameters only, excluding the control inputs.

Hence, in a similar manner to NMPC, NMHE scheme is formulated using a least square function to penalize the deviation of estimated outputs ($\mathbf{h}(\cdot, \cdot)$) from measurements ($\mathbf{z}$). Typically, the estimation horizon contains M measurements ($\mathbf{z}_L, \dots, \mathbf{z}_j$) that are taken at time $t_L < \dots < t_j$, where

the length of the horizon is given by $T_E = t_j - t_L$, and $j - M + 1 \stackrel{\text{def}}{=} L$ is assumed. As the performance of NMHE also relies on the availability of an accurate system model, a mismatch in the form of process noise may deteriorate the optimal estimation solution, which eventually may lead to an unstable closed-loop. To address this issue, a suitable component (arrival cost) is included in the final optimization problem formulation of NMHE (Kühl et al, 2011). Finally, the discrete time dynamic optimization problem to estimate the constrained states ($\hat{x}$) as well as the unknown parameter ($\hat{p}$) at time t_j using the process model $f(\cdot, \cdot, \cdot)$, measurement model $h(\cdot, \cdot)$ and available measurements within the horizon, is of the form:

$$\min_{\hat{x}_k, \hat{p}} \left\{ \left\| \hat{x}_L - \bar{x}_L \right\|_{P_L}^2 + \sum_{k=L}^j \left\| z_k - h(\hat{x}_k, u_k) \right\|_V^2 + \sum_{k=L}^{j-1} \left\| w_k \right\|_W^2 \right\} \quad (10a)$$

$$\text{s.t. } \hat{x}_{k+1} = f_d(\hat{x}_k, u_k, p) + w_k, \quad k = L, \dots, j-1, \quad (10b)$$

$$\hat{x}_{k,\min} \leq \hat{x}_k \leq \hat{x}_{k,\max}, \quad k = L, \dots, j, \quad (10c)$$

$$\hat{p}_{\min} \leq \hat{p} \leq \hat{p}_{\max}, \quad (10d)$$

where w_k represents the added process noise; $\hat{x}_{k,\min} \leq \hat{x}_{k,\max}$ and $\hat{p}_{\min} \leq \hat{p}_{\max}$ specify the lower and upper bounds on the estimated states and parameters, respectively. P_L , V and W are the respective weight matrices that are usually interpreted as the inverse of covariance matrices (Kühl et al, 2011):

$$P_L = Q_0^{-\frac{1}{2}}, \quad V = R^{-\frac{1}{2}}, \quad W = Q^{-\frac{1}{2}}, \quad (11)$$

where Q_0 is the initial covariance matrix (incorporating state and parameter, both), R is the measurement noise covariance matrix, and Q is the process noise covariance matrix. In addition, $\bar{x}_L$ and $\bar{p}_L$ denote the estimated state and parameter values (arrival cost data) at the start of estimation horizon.

In general, the performance of NMHE in terms of its rate of learning is dependent on the choice of estimation window length M , which is mainly problem-specific. That is, in fast dynamical systems like UAVs, we cannot indefinitely increase M as limited computational power is available on-board. Moreover, it is not necessarily true that the estimation accuracy always increases with M , as the plant-model mismatch degrades the significance of model prediction which adversely affects the estimation performance (Kraus et al, 2013). This implies that the selection of a too high value of M for the system in which the unknown parameter (to be estimated) is radically changing, plant-model anomalies may eventually result in a deteriorated overall estimation quality.

5 Implementation

In this implementation, NMPC is designed as a high-level controller that is responsible for tracking a given position trajectory. Based on the current feedback of the other states, the optimized solution for the control input in terms of total thrust and attitude angles is computed. This optimized solution is then passed to the low-level controller as its desired setpoint. Moreover, the low-level controller has a P-PID architecture (implemented in Pixhawk), which comprises of a P controller for attitude and PID controller for angular rate. This low-level control architecture is utilized for each of the three axes (roll, pitch, and yaw) separately, as their dynamics is considered to be decoupled for a standard multirotor UAV. Hence, the combined control vector of the low-level controller can be written as: $u_{P-PID} = [\Omega_1, \Omega_2, \Omega_3, \mu]^T$. The overall control scheme is summarized in a block diagram shown in Fig. 6. Next, we will individually illustrate the design of high-level NMPC and NMHE for the presented framework.

5.1 High-level NMPC Design

The state, control, measurement and parameter vectors for the high-level NMPC are composed of:

$$x_{\text{NMPC}} = [x, y, z, u, v, w]^T, \quad u_{\text{NMPC}} = [\phi, \theta, \psi, F_z]^T, \quad (12a)$$

$$z_{\text{NMPC}} = x_{\text{NMPC}}, \quad p_{\text{NMPC}} = [m, F_{x\text{Dist}}, F_{y\text{Dist}}]^T. \quad (12b)$$

In the parameter vector, one may observe that only the disturbance forces along x - and y - directions, i.e., $F_{x\text{Dist}}$ and $F_{y\text{Dist}}$ are considered. This is because the effect of $F_{z\text{Dist}}$ along z -direction already appears while learning the mass (m) of the UAV. Additionally, the following state and control reference trajectories are selected for the optimization

Fig. 6: Closed-loop control diagram of the proposed instantaneous learning scheme.

Low-level PID gains: $P_{\text{roll}} = 6.5$, $P_{\text{roll-rate}} = 0.1$, $I_{\text{roll-rate}} = 0.05$, $D_{\text{roll-rate}} = 0.002$; $P_{\text{pitch}} = 6.5$, $P_{\text{pitch-rate}} = 0.1$, $I_{\text{pitch-rate}} = 0.05$, $D_{\text{pitch-rate}} = 0.002$; $P_{\text{yaw}} = 4.8$, $P_{\text{yaw-rate}} = 0.15$, $I_{\text{yaw-rate}} = 0.1$, $D_{\text{yaw-rate}} = 0$.

problem of high-level NMPC:

$$\mathbf{x}^{\text{ref}} = \mathbf{x}_{N_c}^{\text{ref}} = [x_r, y_r, z_r, 0, 0, 0]^T, \quad (13a)$$

$$\mathbf{u}^{\text{ref}} = [-0.0414, 0, 0, mg]^T, \quad (13b)$$

where m and g are mass and gravitational constant, listed in Table 1. It is to be noted that for the OCP formulation of high-level NMPC, the parametrization of the nonlinear model (in translation) is also done with respect to p , q and r . Hence, the three rotational rates are fed to high-level NMPC along with the other states, as also illustrated in Fig. 6.

Some input constraints are imposed in the definition of NMPC in order to achieve a stable behavior from the low-level controller:

$$0.5mg \text{ (N)} \leq F_z \leq 1.7mg \text{ (N)}, \quad (14a)$$

$$-20 \text{ (}^\circ\text{)} \leq \phi \leq 20 \text{ (}^\circ\text{)}, \quad (14b)$$

$$-20 \text{ (}^\circ\text{)} \leq \theta \leq 20 \text{ (}^\circ\text{)}, \quad (14c)$$

where m is the maximum takeoff mass (with payload), i.e., $m = 1.9$ kg. Also, the following weight matrices are obtained by trial-and-error:

$$W_x = \text{diag}(22, 25, 20, 1.3, 1.3, 2.0), \quad (15a)$$

$$W_u = \text{diag}(23, 24, 80, 0.012), \quad (15b)$$

$$W_{N_c} = \text{diag}(40, 40, 40, 2, 2, 2), \quad (15c)$$

in order to obtain a smooth response from high-level NMPC. Amongst the above matrices, the terminal weight matrix (W_{N_c}) is weighted more in comparison to the weight matrix for states (W_x). This is a typical way of defining them, where the reason being is to assure the stability of the OCP. It is to be noted that these weight matrices do not imply tuning of the UAV for a particular takeoff mass in a specific scenario. Rather, they are obtained to achieve a smooth response from the UAV for a range of takeoff mass and for all the considered case scenarios in this paper. Furthermore, the prediction window $N_c = 30$ is selected to facilitate the real-time applicability of the proposed control framework.

5.2 NMHE Design

For the considered scenarios in this paper, the main task of NMHE is to estimate online a parameter vector that comprises of UAV mass and two disturbance forces. This parameter vector is made to vary with time by introducing artificial disturbances of different nature in different scenarios. In reference to Fig. 6, the NMHE is designed with the following state, control, measurement and parameter vectors:

$$\mathbf{x}_{\text{NMHE}} = [u, v, w]^T, \quad \mathbf{u}_{\text{NMHE}} = [\phi, \theta, F_z]^T, \quad (16a)$$

$$\mathbf{z}_{\text{NMHE}} = [\mathbf{x}_{\text{NMHE}}, \mathbf{u}_{\text{NMHE}}]^T, \quad \mathbf{p}_{\text{NMHE}} = [m, F_{x\text{Dist}}, F_{y\text{Dist}}]^T. \quad (16b)$$

Again, the three rotational rates are included in the measurements along with states and inputs, in order to solve the underlying OCP, as also done for high-level NMPC.

For the selected tricopter model, the weight matrices P_L , V and W , are chosen to be:

$$P_L = \text{diag}(0.1^2, 0.1^2, 0.1^2, 0.0316^2, 0.1^2, 0.1^2)^{-1/2}, \quad (17a)$$

$$V = \text{diag}(22.36^2, 22.36^2, 22.36^2, 31.62^2, 31.62^2, 14.14^2)^{-1/2}, \quad (17b)$$

$$W = \text{diag}(0.633^2, 0.633^2, 0.633^2, 0.316^2, 0.316^2, 0.316^2)^{-1/2}. \quad (17c)$$

The above values of weight matrices are decided based on calibration experience with the onboard (and external) sensors, incorporating the definitions in (11). In addition, the arrival cost is initialized with the following state and parameter vectors:

$$\bar{\mathbf{x}}_L = [0, 0, 0]^T, \quad \bar{\mathbf{p}}_L = [1.85, 0.1, 0.1]^T. \quad (18)$$

Also, the following constraints are imposed in order to achieve a constrained estimation of the parameter vector:

$$1.35 \text{ (kg)} \leq m \leq 2.0 \text{ (kg)}, \quad (19a)$$

$$-4 \text{ (N)} \leq F_{x\text{Dist}} \leq 4 \text{ (N)}, \quad (19b)$$

$$-4 \text{ (N)} \leq F_{y\text{Dist}} \leq 4 \text{ (N)}, \quad (19c)$$

where the minimum and maximum limits for the disturbance forces are obtained based on experience with the utilized fans. Furthermore, the estimation window length M is selected to be equal to 40, which is more than the prediction horizon length of 30 for high-level NMPC. This is purposely kept in order to realize comparatively slower learning from NMHE.

Remark 2 In the beginning, even longer estimation horizon ($m = 50$) was aimed. However, due to a limited computational power available in Raspberry Pi 3, it was observed that the sampling frequency for NMHE came down to almost 18-Hz. Therefore, in order to maintain the frequency to 30-Hz, the estimation horizon was brought down to 40.

5.3 Solution Technique

Knowing the fact that the optimization problems of NMPC and NMHE have similar forms, both are solved utilizing the same solution technique. Amongst the two most prominent approaches namely, indirect and direct methods, we utilize the latter on account of its several computational advantages over the former. In addition to this classification, the direct methods are further categorized whether being sequential or simultaneous, wherein, a variant of the simultaneous technique, i.e., direct multiple shooting, is favored over the sequential technique (single shooting) in this work. Again, this

selection is due to its several advantages including flexibility in initializing the problem along with parallelizing the algorithm which results in faster convergence properties, especially for unstable systems (Albersmeyer and Diehl, 2010). The resulting discretized OCP reduces to a sequential quadratic program (SQP), which is further solved incorporating the generalized Gauss-Newton (GGN) method. This GGN method is a tailored variant of the classical Newton method which is solved with the help of a special real-time iteration (RTI) scheme proposed in (Diehl et al, 2002). For real-time feasibility, the maximum number of Gauss-Newton iterations are constrained to one within the RTI scheme such that each new OCP is initialized utilizing the solution of the previous. This results in a faster convergence which is crucial for the real-time implementation of NMPC and NMHE on the Raspberry Pi 3 embedded processor.

As a solution platform, we adopt ACADO toolkit (Ariens et al, 2010) which incorporates the direct multiple shooting method and RTI approach for solving (9) and (10). the ACADO toolkit is an open-source C++ based software environment which provides a link to an exhaustive code library for optimization solvers. Firstly, the optimization problems, in terms of system equations and constraints, are defined in a C++ environment and then, the self-contained C codes are obtained utilizing its code generation package (Ariens et al, 2010). Later, these generated C codes can be imported to any computational platform for their final execution.

In our application, we directly import the ACADO generated C codes onboard Raspberry Pi 3. Although Raspberry Pi 3 has low computational power, the optimized nature of generated C codes and our efficiently created ROS (robot operating system) scripts help us achieve a sampling frequency of about 50-Hz for high-level NMPC and about 30-Hz for NMHE. For the feedback of the UAV, an OptiTrack motion capture system, consisting of eight 240 FPS (frames per second) cameras, is utilized which provides the position and attitude data over a local area network. This position and attitude information is given to an estimator (EKF), which subsequently computes the other states including translation and rotational velocities of the UAV. Then, the full-state feedback along with the reference trajectory command ($[x_r, y_r, z_r]^T$) is provided to the high-level NMPC via ROS scripts, over a wireless network. Thereafter, high-level NMPC (running on Raspberry Pi 3) computes the optimal control commands that are fed to the low-level controller (Pixhawk) via serial communication. Finally, the actuator commands are obtained and given to the rotors and tilting servo, as can be visualized from the block diagram in Fig. 7.

Remark 3 It is to be noted that the adopted solution methodology, i.e., direct multiple shooting technique along with a single iteration per sampling solution of the GGN method,

has a limitation of obtaining only the local solution in the vicinity of the previous guess. Hence, not much can be implied about the optimality (local/global) of the obtained solution without performing extensive analysis. Nevertheless, utilizing high sampling rate for the closed-loop and long enough estimation window length positively contribute to the stability of the overall method. Another important point to note here is that the single iteration solution negatively impacts the convergence properties of these methods. That is, with every new reference value, it takes a few iterations before the solution could converge to the actual (sub)optimal point which eventually increases the response time for these methods.

6 Experimental Results

In this section, we present the experimental results of fast NMPC for the high-level position tracking of a 3D-printed tilt-rotor tricopter UAV. The indoor experimental setup including the cameras of the utilized motion capture system is depicted in Fig. 8. In order to thoroughly investigate the efficacy of the proposed learning-based NMPC, three types of experiments incorporating different disturbance sources are devised. Moreover, in all the experiments, the results are obtained for two cases: (i) NMPC without learning (conventional NMPC), and (ii) NMPC with learning (NMPC-NMHE framework), to validate the supremacy of one over the other.

In the subsequent part, we illustrate each of the three experiments in detail. Firstly, a description of the experiment is provided. Thereafter, the implementation details are included which is followed by our interpretation of the obtained results. Different trajectories are selected for these experiments in order to depict the robustness of the framework towards various trajectories. A demonstration video depicting the experiments can be found at: <https://youtu.be/B9I4rPONa44>.

6.1 Case Scenario 1: Payload Dropping

In the first experiment, we tackle a package delivery scenario where a UAV tracks a reference trajectory while dropping packages to their time-based designated locations, as depicted in Fig. 9. The utilized UAV has a total takeoff mass of 1.442 kg with a payload of 457g, which is about 32% of the empty (without payload) mass of the UAV. The drop of this payload implies a massive change in the model parameter (m) that has to be handled by the controller, and hence, justify the use of learning-based framework for this application.

Next, we provide the position tracking results for a time-based circular trajectory of radius 1.5m along x - and y -

Fig. 7: Schematic block-diagram of the real-time implementation.

Notation: $\mathbf{x}_{pos} = [x, y, z]^T$; $\mathbf{x}_{vel} = [u, v, w]^T$; $\mathbf{x}_{att} = [\phi, \theta, \psi]^T$; $\mathbf{x}_{rate} = [p, q, r]^T$.

Fig. 8: Indoor testing environment for the real-time experiments.

directions with a flight speed of 1 m/s. Initially, the state of the UAV is $\mathbf{x}(0) = [0, 0, 1.5, 0, 0, 0]^T$. For both the aforementioned cases (without and with learning), first a complete circular trajectory is tracked by the UAV with maximum takeoff mass and thereafter, the four blocks are dropped at fixed time-intervals in the sequence: 86g $\rightarrow$ 114g $\rightarrow$ 114g $\rightarrow$ 143g. It is to be noted that in this application, since the disturbance by weight dropping is only introduced along z -direction, the contributions of disturbance forces F_{xDist} and F_{yDist} are expected to be negligible. Therefore, they are not estimated here, and the parameter vectors for NMPC and

NMHE are modified to be:

$$\mathbf{p}_{NMPC} = \mathbf{p}_{NMHE} = \mathbf{m}. \quad (20)$$

The overall position tracking performance of both conventional NMPC (without learning) and NMPC-NMHE framework (with learning) are presented together in Figs. 10a and 10b. As visualized from the figures, the z -direction tracking of NMPC-NMHE framework is much precise without any error accumulation with weight drops, in comparison to the conventional NMPC. This effect is anticipated as NMHE in the learning case helps NMPC to adapt itself to the changing dynamics (UAV mass) and hence, the plant-model mismatch (mostly along z -direction) diminishes with time. Moreover, the position tracking Euclidean error and the absolute er-

Fig. 9: Payload drop experiment in Case scenario 1.

ror along z -direction (z_{error}) for both the cases are shown in Fig. 10c. While the mean values of Euclidean error and z_{error} for NMPC without learning case are 0.1598m and 0.1113m, respectively, the learning helps to lower them down as the mean Euclidean error and z_{error} for NMPC-NMHE framework are 0.0962m and 0.0159m, respectively. That is, learning improves the tracking performance by 6.36 cm in terms of Euclidean error and 9.54 cm in terms of z_{error} .

Remark 4 It is to be noted that in this paper, the tracking error is considered to be the distance between the current position and the corresponding closest point on the trajectory.

The control outputs in terms of thrust and attitude angles commanded by the controller within conventional NMPC and NMPC-NMHE framework are presented in Figs. 10e and 10f, respectively. From Figs. 10e and 10f, it is visualized that the NMPC commands for both the cases are well within the constraints specified in (14). Moreover, one may notice a comparatively poor tracking performance in terms of the yaw angle (ψ) response for both the cases. It is emphasized that this is due to the tuning of PIDs along the yaw channel in the low-level controller i.e., within Pixhawk.

Figure 10d depicts the performance of NMHE in estimating the UAV mass, along with its true value. It is observed that the mass estimation always stays within the bounds specified in (19), which eventually illustrates the bounded learning capability of NMHE. This is an important aspect as the stability of NMPC is typically guaranteed for an accurate system model; whereas an unbounded (diverging) estimation of time-varying parameters may eventually destabilize the closed-loop. It is to be noted that, although m is a physical parameter, its estimation by NMHE also accommodates the modeling and operational uncertainties (for instance, discharging of battery) induced during operation. Primarily, it is deduced as an adaptive parameter that enables NMPC to realize an offset-free tracking along the z -direction.

The single Gauss-Newton iteration per sampling instant, performed to solve the optimization problems of NMPC and NMHE would result in a suboptimal solution. Therefore, it becomes necessary to check the optimality of the control framework. To do that, Karush-Kuhn-Tucker (KKT) tolerances are obtained and plotted for the respective NMPC and NMHE in both learning and without learning cases, as shown in Fig. 11a. For a linear system with a least-square objective, these values reduce to zero. However, for the considered tilt-rotor tricopter UAV, these values are anticipated due to the highly nonlinear dynamics along with the added yaw instability (odd number of rotors). Nevertheless, their low and not too drastically varying magnitudes still signify well-defined optimization problems.

Finally, the execution time at each sampling instant for both conventional NMPC and NMPC-NMHE framework is shown in Fig. 11b. While the average execution time for NMPC in without learning case is 4.3 ms, the average execution times for NMPC and NMHE in the case with learning are 4.4 ms and 6.2 ms, respectively. The difference in computation times of NMPC and NMHE within NMPC-NMHE framework is due to the choice of a longer estimation horizon ($M = 40$) over the prediction horizon ($N_e = 30$).

Remark 5 For a shorter horizon, the mass learning is fast which eventually makes NMPC to be aggressive towards the change. On the other hand, for a longer horizon, the NMHE gradually learns the mass parameter and hence, a smooth response is obtained from NMPC. Therefore, selecting an appropriate estimation horizon on Raspberry Pi is a trade-off between the desired rate of learning and the available computational power. In general, the latter is more important due to the minimum sampling frequency required for a sustained closed-loop.

Additionally, in order to statistically support the claim that learning-based NMPC improves the tracking performance in presence of abrupt mass disturbance, experiments with four different levels of payload drop, each repeated 10 times, are performed for both types of NMPCs and their mean Euclidean error values are recorded. Thereafter, a box plot is obtained which is presented in Fig. 12. It is evident that the mean Euclidean error for conventional NMPC increases with the increasing weight drop level, while the mean Euclidean error rise for NMPC-NMHE framework is negligible. This is also according to our expectation because the learning helps NMPC to adapt itself to the changing conditions, and hence, results in a minimal error rise even with the increasing magnitude of disturbance.

6.2 Case Scenario 2: Ground Effect

In the second experiment, we investigate another type of disturbance which is commonly encountered when a UAV flies in proximity to the ground, i.e., when it is within the influence of ground effect. For this scenario, the tricopter UAV is made to track a circular trajectory of radius 1m along x - and y -directions, involving two different heights, as depicted in Fig. 13a. The UAV begins to follow the trajectory at $z = 0.7\text{m}$ with negligible influence of ground effect. Thereafter, while on the circular path, it descends to $z = 0.15\text{m}$, where the ground effect gets dominating. After completing half-a-circle, it climbs back to $z = 0.7\text{m}$ and stays at this height for a while, before going for another round. Furthermore, since the disturbance is only introduced along the z -direction, the influence of disturbance forces ($F_{x\text{Dist}}$, $F_{y\text{Dist}}$) is expected to be negligible in comparison to the ground effect. Therefore, in this application also, these forces are not

(a) 3D view of circular reference trajectory, and the tracked trajectories using conventional NMPC and NMPC-NMHE framework.

(b) Time propagation of reference and actual trajectories.

(c) Time propagation of Euclidean error and error along z-direction.

(d) True values of tricopter mass and its estimation using NMHE.

(e) Controller outputs for conventional NMPC with the actual response of the UAV.

(f) Controller outputs for NMPC-NMHE framework with the actual response of the UAV.

Fig. 10: Case Scenario 1: trajectory tracking control performance of the tilt-rotor tricopter UAV with varying mass, where the vertical magenta lines represent the instants of payload drop. In (d), it is observed that the estimated UAV mass converges to the true value after a transition time of few seconds. Moreover, none of the constraints ever get violated, which implies constrained learning.

Fig. 11: KKT tolerance and execution time for the payload drop experiment. It is to be noted that in the interest of space, these plots are only provided for Case scenario 1.

Fig. 12: Mean Euclidean error for four levels of abrupt weight drops, where 10 tests for each weight level are performed.

estimated, and the corresponding NMPC and NMHE parameter vectors are:

$$\mathbf{p}_{\text{NMPC}} = \mathbf{p}_{\text{NMHE}} = \mathbf{m}. \quad (21)$$

The position tracking performance for both without and with learning cases, in presence of ground effect can be seen in Figs. 13a and 13b. As expected, the learning-based NMPC dominates the conventional NMPC in terms of z -direction tracking. The improvement is better visualized from Fig. 13c, where the Euclidean error and z_{error} are plotted. As observed in Fig. 13c (particularly z_{error} plot), there lies an offset along the z -direction for conventional NMPC, due to which its z_{error} always stays more than that for NMPC-NMHE framework. Although there are two instances (around 15s and 30s) where the z_{error} for NMPC-NMHE framework becomes more, this is the part when tricopter is climbing and hence, can be associated with the transient response of the UAV. That is, at the instant

when tricopter starts to climb, its estimated (effective) mass within NMPC-NMHE framework is less, and hence, the tracking error instantaneously rises until the new true mass is estimated by NMHE. Moreover, in the experiments with ground effect, the mean values for Euclidean error and z_{error} in conventional NMPC case are 0.1852m and 0.1091m, respectively, while for NMPC-NMHE framework the mean error values are 0.1380m and 0.0459m, respectively. Hence, the learning resulted in an improvement of 4.72 cm in terms of Euclidean error and 6.32 cm in terms of z_{error} .

Next, the control outputs for conventional NMPC and NMPC-NMHE framework are presented in Figs. 13e and 13f, respectively. From these figures, it is evident that NMPC commands for both the cases are well within the specified constraints. Moreover, comparably poor tracking of the commanded yaw by the UAV can again be explained, happening due to a conservative tuning of low-level PIDs along the yaw channel.

The mass estimation performance of NMHE in NMPC-NMHE framework is depicted in Fig. 13d. The estimated value always stays restricted within the specified bounds that again showcases the bounded learning capability of NMHE. In addition, the estimated mass value varies in an intuitive pattern along the trajectory. That is, when the UAV is flying under the influence of ground effect, the effective mass is expected to decrease as also seen in sections II and IV of Fig. 13d, and vice versa for the UAV flying away from the ground.

Finally, the average execution time for NMPC in without learning case is 4.21 ms, while the average execution times for NMPC and NMHE in the case with learning are 4.33 ms and 7.591 ms, respectively.

(a) 3D view of circular reference trajectory, and the tracked trajectories using conventional NMPC and NMPC-NMHE framework.

(b) Time propagation of reference and actual trajectories.

(c) Time propagation of Euclidean error and error along z -direction.

(d) Tricopter mass estimation using NMHE.

(e) Controller outputs for conventional NMPC with the actual response of the UAV.

(f) Controller outputs for NMPC-NMHE framework with the actual response of the UAV.

Fig. 13: Case Scenario 2: trajectory tracking control performance of the tilt-rotor tricopter UAV in presence of ground effect, where the vertical magenta lines in (c) and (d) represent sections with the same z -position, such that sections I, III and V are with $z = 0.7\text{m}$, and sections II and IV are with $z = 0.15\text{m}$. Additionally, in (d), it is observed that the tricopter mass decreases as it hovers close to the ground which is also intuitively consistent.

6.3 Case Scenario 3: Wind-gust Disturbance

In the third experiment, we mimic one of the most commonly occurring disturbance, i.e., wind-gust, for a UAV flying outdoor. Although all our experiments are performed indoors, the main goal of this work is to bring the UAV from an engineered (well-known) indoor environment to a less known outdoor world, as mentioned previously. Also, in order to achieve an exhaustive investigation in this paper, studying the effects of wind-gust on a flying UAV and evaluating the improvements that a learning-based framework can bring, are mandatory tasks. Therefore, in this scenario, artificial wind-gust disturbances are introduced with the help of two industrial fans that are placed perpendicular to each other at a distance of 2.5m from the origin. Overall, they generate disturbances of magnitude 3.3-3.7 m/s (each fan) along $+x$ and $-y$ directions, as can be visualized from Fig. 8. For a better judgment on the learning ability of NMPC-NMHE framework, the experiments are conducted with two trajectories namely, slanted circle, and hover at the origin with sinusoidally varying z -position. Moreover, in this experiment, since the disturbances are explicitly introduced along x - and y -directions, the magnitudes of disturbance forces (F_{xDist} , F_{yDist}) are expected to get substantial. Hence, they are also estimated along with the mass of the UAV:

$$\mathbf{p}_{NMPC} = \mathbf{p}_{NMHE} = [m, F_{xDist}, F_{yDist}]^T. \quad (22)$$

Next, we sequentially present the trajectory tracking results for slanted circle and hover trajectories.

Case Scenario 3a – Slanted Circular Trajectory:

The position tracking performance of NMPC for a slanted circular trajectory of radius 1m, in without and with learning cases can be seen in Figs. 14a and 14b. To further analyze tracking accuracy, the Euclidean error is also plotted for both the cases and is presented in Fig. 14c. From Fig. 14c, it is evident that the Euclidean error for NMPC-NMHE framework always stays below than that for conventional NMPC. In addition, the mean Euclidean error values for without and with learning cases are 0.2078m and 0.1515m, respectively, which eventually implies an improvement of 5.63 cm due to learning.

Remark 6 Looking at these error values, one may question the magnitude of improvement that a learning-based framework brings to this application. Regarding this doubt, we would like to emphasize that when the UAV encounters a wind-gust disturbance, there is a time-lapse before which the magnitude of the learned disturbance force is fed to NMPC and it can perform the necessary corrections. However, by the time NMPC receives the learned disturbance

values, the UAV proceeds further on its course, making the estimated values obsolete for the current corrections. Hence, there lies a time-lag between the estimation and corrective action taken by the controller, thereby making learning less effective within the scope of this trajectory. Nevertheless, in order to validate the proficiency of learning, these experiments are performed again for a hover with varying z -position trajectory, the results of which are discussed later.

For the sake of completeness, control outputs for conventional NMPC and NMPC-NMHE framework for a slanted circular trajectory are presented in Figs. 14e and 14f, respectively. From these figures, it is again evident that the NMPC commands for both the cases are well within the specified bounds.

Figure 14d depicts the performance of NMHE in estimating the UAV mass along with the introduced disturbance forces in NMPC-NMHE framework. The constrained estimation ability of NMHE is again validated here, as none of the specified bounds are ever violated, even when the estimated values for disturbance forces reached the bounds.

Remark 7 Since the disturbance forces are induced via same wind speed for both the fans, the estimator is expected to produce less oscillating estimation values. However, it is pointed out that due to the circular motion of the UAV, the effective disturbance forces acting on the UAV vary with time. This assertion is validated by the estimation results of the hover trajectory that are presented later in this section.

Furthermore, the average execution time for NMPC in without learning case is 4.212 ms, while the average execution times for NMPC and NMHE in the case with learning are 4.92 ms and 6.694 ms, respectively.

Case Scenario 3b – Hover Trajectory:

As asserted before, due to the time-varying nature of the wind-gust disturbance while flying a circular trajectory, NMHE does not have enough time to learn the true disturbance values. As a result, only a limited performance improvement is observed. Therefore, in order to validate the efficacy of learning in dealing with the induced wind-gust disturbances, these experiments are performed again for a hover trajectory, involving a motion only along the z -direction. While selecting the hover trajectory, it is expected that the stationary (along x - and y -directions) nature of this trajectory would provide ample time to NMHE for learning the disturbance forces, such that, a significant improvement in tracking would be observed.

The position tracking results for a hover with changing z -position reference by both types of NMPCs can be seen in Figs. 15a and 15b. As expected, there lies a steady offset along $+x$ - and $-y$ -directions for conventional NMPC, while

(a) 3D view of slanted circular reference trajectory, and the tracked trajectories using conventional NMPC and NMPC-NMHE framework.

(b) Time propagation of reference and actual trajectories.

(c) Time propagation of Euclidean error.

(d) Tricopter mass and disturbance forces estimation using NMHE.

(e) Controller outputs for conventional NMPC with the actual response of the UAV.

(f) Controller outputs for NMPC-NMHE framework with the actual response of the UAV.

Fig. 14: Case Scenario 3a: trajectory tracking control performance of the tilt-rotor tricopter UAV in presence of wind-gust disturbance for a slanted circular reference. In (d), it is observed that none of the constraints are violated that again validates the safe learning capability of NMHE.

the NMPC within NMPC-NMHE framework compensates it. This is mainly due to the learning ability of NMHE,

which makes NMPC account for the experienced disturbance forces. Moreover, the corrective actions of learning-

based NMPC are more visible in Fig. 15c, where the Euclidean errors for both without and with learning NMPCs are presented. In addition, the mean Euclidean error values for without and with learning cases are 0.1760m and 0.1047m, respectively, which eventually showcases an improvement of 7.13 cm brought by learning for this scenario. Furthermore, the estimation results of NMHE within NMPC-NMHE framework are shown in Fig. 15d. One may observe a comparatively less oscillating estimation of disturbance forces for this hover trajectory, which is in contrast with the estimation results for the previous trajectory. As a summary, the mean Euclidean and absolute z -position error values for all the aforementioned test case scenarios, obtained with both types of NMPCs are listed in Table 2.

Finally, in order to further support the claim that learning-based NMPC improves the tracking performance in presence of wind-gust disturbance, 10 experiments each with three different levels of wind speeds are performed for both types of NMPCs and their mean Euclidean error values are recorded. Thereafter, a box plot is obtained which is presented in Fig. 16. It is evident from Fig. 16 that the mean Euclidean error for both type of NMPCs increase with the increasing wind speed, which is also intuitive. However, the error rise for NMPC-NMHE framework with increasing wind speed is minimal compared to the one for conventional NMPC. This is also in accordance with our expectation because the learning helps NMPC to adapt itself to the changing conditions, and hence, results in a slower error rise even with the increasing magnitude of disturbances.

Remark 8 One may think that the wind speeds which are utilized for testing may not be too strong. However, we would like to emphasize that these speeds are strong enough to employ a significant gust-force on the considered UAV. This is due to the high ratio of the applied wind-gust disturbance magnitude in comparison to the UAV size/mass. Moreover, it is to be noted that the closed-loop framework should be able to handle any wind-gust magnitude, provided that the UAV actuators (rotors) can handle the disturbance force and the NMHE is designed with the appropriate constraint values.

Table 2: Mean Euclidean and absolute z -position errors for the three case scenarios. (N.A.: not applicable)

	Euclidean error (cm)		z -position error (cm)	
	NMPC	NMPC-NMHE	NMPC	NMPC-NMHE
Case 1	17.72	15.93	5.61	1.09
Case 2	18.52	13.80	10.91	4.59
Case 3a	20.78	15.15	N.A.	N.A.
Case 3b	17.70	10.47	N.A.	N.A.

7 Conclusion

In this paper, we have developed an instantaneous learning-based NMPC for the trajectory tracking of a 3D-printed tilt-rotor tricopter UAV without violating the system constraints, resulting in safe learning. The main motivation is to develop an aerial robot that can learn from its past interactions with the environment, and thus, can perform sufficiently well in unknown working conditions for which the aerial robot has not been trained. Within the utilized learning-based framework, the NMPC is designed as the high-level controller for the position tracking of the UAV, and the NMHE is employed to simultaneously learn the time-varying model parameters in an instantaneous manner. Thanks to its learning capability, the NMPC can adapt itself to the changing working conditions, and hence, performs satisfactorily well in any unknown working condition. The efficacy of the proposed learning-based framework has been investigated for three case scenarios, where disturbances in the form of mass variation, ground effect, and wind-gust, are explicitly induced. On account of learning, the z -direction tracking improvements of 85.7% and 57.9% have been observed for the case scenarios 1 and 2, respectively. In case scenario 3, firstly with a slanted circular trajectory, the achieved improvement of 27.1% in terms of Euclidean error has been found insufficient in portraying the true capability of the learning-based framework. Therefore, the experiments with wind-gust disturbances have been repeated for a hover with varying z -position trajectory, which finally resulted in a Euclidean error improvement of 40.5% with the help of learning. Additionally, the performed statistical analysis validated a consistent improvement by learning-based NMPC when compared to conventional NMPC for the payload drop and wind-gust disturbance scenarios. What is more, owing to the fast C++ based execution of NMPC and NMHE codes, a complete onboard implementation of the presented closed-loop could be realized on a low-cost embedded processor. Designed with more adaptive control structures – also with the help of artificial intelligence and machine learning methods –, it is not so far-fetched that the robots of the future could learn from their own experience, interact with environment, and eventually adapt themselves to different working conditions.

Acknowledgements This work was equally financially supported by the National Research Foundation, Prime Ministers Office, Singapore under its Medium-Sized Centre funding scheme, and the Aarhus University, Department of Engineering (28173). In addition, the authors would like to acknowledge Siddharth Patel, Lee Ying Jun Wilson and Suraj Ravindrababu for their support during the real-time experiments.

(a) 3D view of hover reference trajectory, and the tracked trajectories using conventional NMPC and NMPC-NMHE framework.

(b) Time propagation of reference and actual trajectories.

(c) Time propagation of Euclidean error.

(d) Tricopter mass and disturbance forces estimation using NMHE.

Fig. 15: Case Scenario 3b: trajectory tracking control performance of the tilt-rotor tricopter UAV in presence of wind-gust disturbance for a hover with varying z -position reference. It is to be noted that in (d), smoother estimation performance is observed in contrast with Fig. 14d.

Fig. 16: Mean Euclidean error for three levels of wind speed, where 10 tests for each speed level are performed. Range of wind speeds for various levels are, Level 1: 2.4 – 3.0 m/s; Level 2: 3.3 – 3.7 m/s; Level 3: 3.8 – 4.2 m/s.

References

- Albersmeyer J, Diehl M (2010) The lifted newton method and its application in optimization. *SIAM Journal on Optimization* 20(3):1655–1684, DOI 10.1137/080724885, URL <https://doi.org/10.1137/080724885>, <https://doi.org/10.1137/080724885>
- Alexis K, Nikolakopoulos G, Tzes A (2011) Switching model predictive attitude control for a quadrotor helicopter subject to atmospheric disturbances. *Control Engineering Practice* 19(10):1195 – 1207, DOI <https://doi.org/10.1016/j.conengprac.2011.06.010>, URL <http://www.sciencedirect.com/science/article/pii/S0967066111001262>
- Alexis K, Papachristos C, Siegwart R, Tzes A (2016) Robust Model Predictive Flight Control of Unmanned Rotorcrafts. *Journal of Intelligent & Robotic Systems* 81(3):443–469

- Amazon (2017) Amazon prime air. URL <https://www.amazon.com/Amazon-Prime-Air/b?node=8037720011>, [accessed 4-September-2017]
- Ariens D, Houska B, Ferreau H, Logist F (2010) ACADO: Toolkit for Automatic Control and Dynamic Optimization. Optimization in Engineering Center (OPTEC), 1st edn, <http://www.acadotoolkit.org/>
- Ataka A, Tnunay H, Inovan R, Abdurrohman M, Preas-tianto H, Cahyadi AI, Yamamoto Y (2013) Controllability and observability analysis of the gain scheduling based linearization for UAV quadrotor. In: 2013 International Conference on Robotics, Biomimetics, Intelligent Computational Systems, pp 212–218, DOI 10.1109/ROBIONETICS.2013.6743606
- Bemporad A, Pascucci C, Rocchi C (2009) Hierarchical and hybrid model predictive control of quadcopter air vehicles. IFAC Proceedings Volumes 42(17):14 – 19, DOI <https://doi.org/10.3182/20090916-3-ES-3003.00004>, URL <http://www.sciencedirect.com/science/article/pii/S1474667015307308>, 3rd IFAC Conference on Analysis and Design of Hybrid Systems
- Bircher A, Kamel M, Alexis K, Oleynikova H, Siegwart R (2016) Receding horizon “next-best-view” planner for 3d exploration. In: 2016 IEEE International Conference on Robotics and Automation (ICRA), pp 1462–1468, DOI 10.1109/ICRA.2016.7487281
- Bouabdallah S (2007) Design and control of quadrotors with application to autonomous flying. PhD dissertation, EPFL
- Bouabdallah S, Noth A, Siegwart R (2004) PID vs LQ control techniques applied to an indoor micro quadrotor. In: 2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (IEEE Cat. No.04CH37566), vol 3, pp 2451–2456 vol.3
- Bouffard P, Aswani A, Tomlin C (2012) Learning-based model predictive control on a quadrotor: Onboard implementation and experimental results. In: 2012 IEEE International Conference on Robotics and Automation (ICRA), pp 279–284, DOI 10.1109/ICRA.2012.6225035
- Cao G, Lai EMK, Alam F (2016) Gaussian process model predictive control of unmanned quadrotors. In: 2016 2nd International Conference on Control, Automation and Robotics (ICCAR), pp 200–206, DOI 10.1109/ICCAR.2016.7486726
- Chen J, Liu H, Zheng J, Lv M, Yan B, Hu X, Gao Y (2016) Damage degree evaluation of earthquake area using UAV aerial image. International Journal of Aerospace Engineering 2016
- Chowdhary G, Mühlegg M, How JP, Holzapfel F (2013a) Concurrent learning adaptive model predictive control, Springer Berlin Heidelberg, Berlin, Heidelberg, pp 29–47. DOI 10.1007/978-3-642-38253-6_3, URL https://doi.org/10.1007/978-3-642-38253-6_3
- Chowdhary G, Mhlegg M, How JP, Holzapfel F (2013b) A concurrent learning adaptive-optimal control architecture for nonlinear systems. In: 52nd IEEE Conference on Decision and Control, pp 868–873, DOI 10.1109/CDC.2013.6759991
- Dentler J, Kannan S, Bezzaoucha S, Olivares-Mendez MA, Voos H (2018) Model predictive cooperative localization control of multiple uavs using potential function sensor constraints. Autonomous Robots DOI 10.1007/s10514-018-9711-z, URL <https://doi.org/10.1007/s10514-018-9711-z>
- Diehl M, Uslu I, Findeisen R, Schwarzkopf S, Allgöwer F, Bock HG, Bürner T, Gilles ED, Kienle A, Schlöder JP, Stein E (2001) Real-Time optimization for large scale processes: Nonlinear model predictive control of a high purity distillation column, Springer Berlin Heidelberg, Berlin, Heidelberg, pp 363–383. DOI 10.1007/978-3-662-04331-8_20, URL https://doi.org/10.1007/978-3-662-04331-8_20
- Diehl M, Bock H, Schlöder JP, Findeisen R, Nagy Z, Allgöwer F (2002) Real-time optimization and nonlinear model predictive control of processes governed by differential-algebraic equations. Journal of Process Control 12(4):577 – 585, DOI [https://doi.org/10.1016/S0959-1524\(01\)00023-3](https://doi.org/10.1016/S0959-1524(01)00023-3), URL <http://www.sciencedirect.com/science/article/pii/S0959152401000233>
- Eren U, Prach A, Koçer BB, Raković SV, Kayacan E, Açkmeşe B (2017) Model predictive control in aerospace systems: Current state and opportunities. Journal of Guidance, Control, and Dynamics 40(7):1541–1566, DOI 10.2514/1.G002507, URL <https://doi.org/10.2514/1.G002507>
- Garcia GA, Keshmiri S (2013) Online artificial neural network model-based nonlinear model predictive controller for the meridian UAS. International Journal of Robust and Nonlinear Control 23(15):1657–1681, DOI 10.1002/rnc.3037, URL <http://dx.doi.org/10.1002/rnc.3037>
- Garimella G, Kobilarov M (2015) Towards model-predictive control for aerial pick-and-place. In: 2015 IEEE International Conference on Robotics and Automation (ICRA), pp 4692–4697, DOI 10.1109/ICRA.2015.7139850
- Grüne L (2012) NMPC without terminal constraints. IFAC Proceedings Volumes 45(17):1–13
- Kayacan E, Kayacan E, Ramon H, Saeys W (2014a) Distributed nonlinear model predictive control of an autonomous tractor-trailer system. Mechatronics 24(8):926 – 933
- Kayacan E, Kayacan E, Ramon H, Saeys W (2014b) Nonlinear modeling and identification of an autonomous tractor-trailer system. Computers and Electronics in Agriculture 106:1 – 10, DOI <http://dx.doi.org/10.1016/j.compag.>

- 2014.05.002
- Kayacan E, Kayacan E, Ramon H, Saeys W (2015a) Learning in centralized nonlinear model predictive control: Application to an autonomous tractor-trailer system. *IEEE Transactions on Control Systems Technology* 23(1):197–205
- Kayacan E, Kayacan E, Ramon H, Saeys W (2015b) Towards agrobots: Identification of the yaw dynamics and trajectory tracking of an autonomous tractor. *Computers and Electronics in Agriculture* 115:78–87
- Kayacan E, Peschel JM, Kayacan E (2016) Centralized, decentralized and distributed nonlinear model predictive control of a tractor-trailer system: A comparative study. In: 2016 American Control Conference (ACC), pp 4403–4408, DOI 10.1109/ACC.2016.7525615
- Kraus T, Ferreau H, Kayacan E, Ramon H, Baerdemaeker JD, Diehl M, Saeys W (2013) Moving horizon estimation and nonlinear model predictive control for autonomous agricultural vehicles. *Computers and Electronics in Agriculture* 98:25–33
- Kühl P, Diehl M, Kraus T, Schlöder JP, Bock HG (2011) A real-time algorithm for moving horizon state and parameter estimation. *Computers & Chemical Engineering* 35(1):71–83
- Lee WYJ, Mehndiratta M, Kayacan E (2018) Fly without borders with additive manufacturing: a microscale tilt-rotor tricopter design. In: *Proceedings of the 3rd International Conference on Progress in Additive Manufacturing (Pro-AM 2018)*, pp 256–261, DOI 10.25341/D43K5G
- Li B, Zhou W, Sun J, Wen C, Chen C (2018) Model predictive control for path tracking of a VTOL tailsitter UAV in an HIL simulation environment. In: 2018 AIAA Modeling and Simulation Technologies Conference, American Institute of Aeronautics and Astronautics, p 1919
- Mayne D, Rawlings J, Rao C, Scokaert P (2000) Constrained model predictive control: Stability and optimality. *Automatica* 36(6):789–814
- Mehndiratta M, Kayacan E (2017) Receding horizon control of a 3 DOF helicopter using online estimation of aerodynamic parameters. *Proceedings of the Institution of Mechanical Engineers, Part G: Journal of Aerospace Engineering* DOI 10.1177/0954410017703414, URL <http://dx.doi.org/10.1177/0954410017703414>
- Mehndiratta M, Kayacan E (2018a) Online learning-based receding horizon control of tilt-rotor tricopter: A cascade implementation. In: 2018 American Control Conference (ACC), pp 1–6
- Mehndiratta M, Kayacan E (2018b) Reconfigurable fault-tolerant nmpe for y6 coaxial tricopter with complete loss of one rotor. In: 2018 IEEE Conference on Control Technology and Applications (CCTA), pp 774–780, DOI 10.1109/CCTA.2018.8511444
- Mehndiratta M, Kayacan E, Kayacan E (2018) A simple learning strategy for feedback linearization control of aerial package delivery robot. In: 2018 IEEE Conference on Control Technology and Applications (CCTA), pp 361–367, DOI 10.1109/CCTA.2018.8511485
- Mehndiratta M, Kayacan E, Patel S, Kayacan E, Chowdhary G (2019) Learning-based fast nonlinear model predictive control for custom-made 3D printed ground and aerial robots. *Control Engineering* DOI 10.1007/978-3-319-77489-3
- Mellinger D, Lindsey Q, Shomin M, Kumar V (2011) Design, modeling, estimation and control for aerial grasping and manipulation. In: 2011 IEEE/RSJ International Conference on Intelligent Robots and Systems, pp 2668–2673, DOI 10.1109/IROS.2011.6094871
- Mellinger D, Michael N, Kumar V (2012) Trajectory generation and control for precise aggressive maneuvers with quadrotors. *The International Journal of Robotics Research* 31(5):664–674, DOI 10.1177/0278364911434236, URL <https://doi.org/10.1177/0278364911434236>
- Nicolao GD, Magni L, Scattolini R (1998) Stabilizing receding-horizon control of nonlinear time-varying systems. *IEEE Transactions on Automatic Control* 43(7):1030–1036
- Ostafew CJ, Schoellig AP, Barfoot TD, Collier J (2016) Learning-based nonlinear model predictive control to improve vision-based mobile robot path tracking. *Journal of Field Robotics* 33(1):133–152, DOI 10.1002/rob.21587, URL <http://dx.doi.org/10.1002/rob.21587>
- Prach A, Kayacan E (2017) An MPC-based position controller for a tilt-rotor tricopter VTOL UAV. *Optimal Control Applications and Methods* DOI 10.1002/oca.2350, URL <http://dx.doi.org/10.1002/oca.2350>
- Qi J, Song D, Shang H, Wang N, Hua C, Wu C, Qi X, Han J (2016) Search and Rescue Rotary-Wing UAV and Its Application to the Lushan Ms 7.0 Earthquake. *Journal of Field Robotics* 33(3):290–321
- Rajappa S, Blthoff H, Stegagno P (2017) Design and implementation of a novel architecture for physical human-UAV interaction. *The International Journal of Robotics Research* 36(5-7):800–819, DOI 10.1177/0278364917708038, URL <https://doi.org/10.1177/0278364917708038>
- Schoellig AP, Mueller FL, D’Andrea R (2012) Optimization-based iterative learning for precise quadcopter trajectory tracking. *Autonomous Robots* 33(1):103–127, DOI 10.1007/s10514-012-9283-2, URL <https://doi.org/10.1007/s10514-012-9283-2>
- Seo H, Kim S, Kim HJ (2017) Aerial grasping of cylindrical object using visual servoing based on stochastic model

- predictive control. In: 2017 IEEE International Conference on Robotics and Automation (ICRA), pp 6362–6368, DOI 10.1109/ICRA.2017.7989751
- Shin J, Kim HJ, Park S, Kim Y (2010) Model predictive flight control using adaptive support vector regression. *Neurocomputing* 73(4):1031 – 1037, DOI <https://doi.org/10.1016/j.neucom.2009.10.002>, URL <http://www.sciencedirect.com/science/article/pii/S0925231209003774>, bayesian Networks / Design and Application of Neural Networks and Intelligent Learning Systems (KES 2008 / Bio-inspired Computing: Theories and Applications (BIC-TA 2007)
- Szafir D, Mutlu B, Fong T (2017) Designing planning and control interfaces to support user collaboration with flying robots. *The International Journal of Robotics Research* 36(5-7):514–542, DOI 10.1177/0278364916688256, URL <https://doi.org/10.1177/0278364916688256>
- Vukov M, Gros S, Horn G, Frison G, Geebelen K, Jørgensen J, Swevers J, Diehl M (2015) Real-time nonlinear MPC and MHE for a large-scale mechatronic application. *Control Engineering Practice* 45:64 – 78