Nonlinear Model Predictive Control with Enhanced Actuator Model for Multi-Rotor Aerial Vehicles with Generic Designs

Davide Bicego, Jacopo Mazzetto, Ruggero Carli, Marcello Farina, Antonio Franchi

Noname manuscript No.
(will be inserted by the editor)

Preprint version, final version at https://www.springer.com/journal/10846

Accepted for publication at the Springer Journal of Intelligent & Robotic Systems, 2020

Nonlinear Model Predictive Control with Enhanced Actuator Model for Multi-Rotor Aerial Vehicles with Generic Designs

Davide Bicego · Jacopo Mazzetto · Ruggero Carli · Marcello Farina · Antonio Franchi

Received: date / Accepted: date

Abstract In this paper, we propose, discuss, and validate an online Nonlinear Model Predictive Control (NMPC) method for multi-rotor aerial systems with arbitrarily positioned and oriented rotors which simultaneously addresses the local reference trajectory planning and tracking problems. This work brings into question some common modeling and control design choices that are typically adopted to guarantee robustness and reliability but which may severely limit the attainable performance. Unlike most of state of the art works, the proposed method takes advantages of a unified nonlinear model which aims to describe the whole robot dynamics by explicitly including a realistic physical description of the actuator dynamics and limitations. As a matter of fact, our solution does not resort to common simplifications such as: 1) linear model approximation, 2) cascaded control paradigm used to decouple the translational and the rotational dynamics of the rigid body, 3) use of low-level reactive trackers for the stabilization of the internal loop, and 4) unconstrained optimization resolution or use of fictitious constraints. More in detail, we consider as control inputs the derivatives of the propeller forces and propose a novel method to suitably identify the actuator limitations by leveraging experimental data. Differently from previous approaches, the constraints of the optimization problem are defined only by the real physics of the actuators, avoiding conservative – and often not physical – input/state saturations which are present, e.g., in cascaded approaches. The control algorithm is implemented using a state-of-the-art Real Time Iteration (RTI) scheme with partial sensitivity update method. The performances of the control system are finally validated by means of real-time simulations and in real experiments, with a large spectrum of heterogeneous multi-rotor systems: an *under-actuated* quadrotor, a *fully-actuated* hexarotor, a multi-rotor with *orientable* propellers, and a multi-rotor with an unexpected *rotor failure*. To the best of our knowledge, this is the first time that a predictive controller framework with all the valuable aforementioned features is presented and extensively validated in real-time experiments and simulations.

Keywords Model Predictive Control · Multi-Rotor Aerial Vehicles · Multi-Directional Thrust · Actuator Constraints

1 Introduction

In the last decade, thanks to the development of both new hardware technologies and software algorithms, the employment of Multi-Rotor Aerial Vehicles (MRAVs) has significantly spread across a wide set of challenging real-life applications, thanks to their vertical take-off and landing (VTOL) and hovering capabilities, their agility, relatively compact structure, good robustness, and low cost. Classical multi-rotor platforms with *under-actuated* dynamics (e.g., the popular quadrotors), have been extensively studied by the scientific community and widely employed in *contact-less* civil applications such as aerial photography, visual inspection of infrastructures, area patrolling, crop monitoring, and urban search and rescue (USAR) missions [1]. The total thrust direction in the body frame of these platforms is fixed and a re-orientation of the robot chassis is needed to continuously steer the exerted force towards the desired direction. We refer to the vehicles in this class as unidirectional-thrust (UDT) aerial vehicles.

This research was partially supported by the cooperation program INTERREG Deutschland-Nederland as part of the SPECTORS project number 143081 and by the European Unions Horizon 2020 research and innovation program grant agreement ID: 871479 AERIAL-CORE.

D.B. and A.F. are with the Robotics and Mechatronics group, University of Twente, Enschede, The Netherlands and with LAAS-CNRS, Université de Toulouse, CNRS, Toulouse, France. J.M. and R.C. are with Department of Information Engineering, University of Padova, Padova, Italy. M.F. is with Dipartimento di Elettronica, Informazione e Bioingegneria, Politecnico of Milano, Milan, Italy. Corresponding author: D.B., e-mail: d.bicego@utwente.nl.

On the other hand, recent platforms characterized by particular actuator arrangements can exploit the multidirectional-thrust (MDT) capability, i.e., the possibility to exert forces in more than one direction without the need to re-orient their body frame, allowing to partially decouple the robot rotational dynamics from the translational one. A subset of this class is represented by the so-called *fully-actuated* systems, for which the control force can be varied in all directions, disregarding the actuator constraints. Vehicles of this kind have been demonstrated to be particularly suitable for the accomplishment of aerial physical interactions tasks [2,3], i.e., operations which require an *active contact* and a consequent exchange of energy between the robots and the surrounding environment. Examples of such operations are grasping, transportation, and manipulation of loads, contact-based inspection tasks, and building/decommissioning of structures.

Many different control strategies for MRAVs have been designed for trajectory tracking. The most common controllers implemented on these systems are PIDs (i.e., Proportional, Integrative and Derivative) designed based on models, either linearized around the hovering condition as in [24], or obtained with feedback linearization as in [25–27]. Other control methods applied to MRAV include, but are not limited to, adaptive control [28], back-stepping and sliding-mode [29]. The interested reader is addressed to [30] for a detailed overview about available control strategies for underactuated MRAVs, while an extension of [25] to the fully-actuated case has been proposed in our previous work [31]. The main limitations of the mentioned algorithms are: (i) they are *not predictive*, in the sense that the control input at any time instant is not computed with the objective of optimizing the system performance on a future time horizon, possibly based on a reference motion trajectory; (ii) they are not able to enforce the fulfillment of limitations on input and state variables, which might be crucial for safety reasons.

In the last decades, intense research has been devoted to the development, testing, and implementation of Model Predictive Control (MPC), a model-based optimization-based predictive control method which has gained large popularity especially in the process and chemical industries. More recently, thanks to the growing availability of increasingly efficient embedded computers, the popularity of MPC is broadening to safety and time-critical applications with fast dynamics, e.g., in the automotive and robotic fields. MPC is nowadays theoretically well founded and its popularity is mainly related to the following facts. First, it is able to optimize, in a predictive fashion, the system behavior on a given future time horizon based on the system model. Also, in view of the fact that (at least in its most common implementation) it is based on the iterative solution of a constrained optimal control problem (OCP), it allows to enforce dynamic constraints on the state and the inputs of a physical system. Furthermore, since the related OCP is solved at each sampling instant as new state measurements get available, it is able to mitigate for possible model perturbations.

Regarding the application of MPC to MRAVs, several notable works have been done in the past few years. On the one hand, some papers tackle the problems of *offline* generating (by solving a suitable OCP) a reference trajectory, feasible with respect to (w.r.t.) the state limits of the system while avoiding possible fixed obstacles, e.g., [16, 18–20]. Other references to MPC in this perspective can be found in a recent review of motion planning methods for swarms of aerial robots [32]. Other works, instead, are devoted to closed-loop schemes, that allow for stabilization of the vehicle dynamics and possibly for local trajectory planning.

Here we will focus on the latter class. In this framework, cascaded control schemes are very common, that rely on the decoupling between the translational and the rotational dynamics of the rigid body. In the majority of the works that rely on this approach, e.g., [5, 7, 9, 10, 14], MPC is used for position control, while the inner-loop attitude control task is obtained using unconstrained regulators (e.g., Lyapunov-based, PIDs, etc.). On the contrary, in [13, 15], the authors employ MPC for control of the inner rotational loop. In either ways, the common strategy is to stabilize the rotational dynamics in an inner loop and to use the rotation configuration (or the angular velocity) as a virtual input commanded by the outer position-control loop. However, cascaded control methods do not allow to exploit the potentialities of the vehicles at their best, in our opinion. Indeed, the problem with this decoupled approach is the introduction of *fictitious* (non-real from a physical point of view) constraints in the virtual inputs, i.e., in the state variables that represent the interface between the two nested controlled systems. As a matter of fact, any constraint imposed on state variables such as the linear velocity, acceleration, jerk, snap, or on the orientation (e.g., Euler angles) and the angular velocity, constitutes a heuristic limitation which does not model accurately the real physical constraints of the real system. On the other hand, in our work, the complete system dynamics is modeled in a non-cascaded way and only the limitations on the individual motor thrust forces and their rates are enforced.

As a matter of fact, the only constraints that play the major role in the platform dynamics are the maximum and minimum torques that can be attained by the motors which drive the propellers. Such limits cause a maximum speed (mainly due to air drag), a minimum speed (mainly due to electronic reactions), a maximum acceleration (mainly due to motor/propeller inertia), and a maximum deceleration (mainly due to nonlinear active breaking). Any simplification which replaces such real constraints with fictitious constraints in the configuration/state of the platform results, unavoidably, in a reduced control performance w.r.t. the real dynamic potential of the robot. In support for the need for a "whole system" control, a few recent works, e.g., [10–12, 17, 21],

Accepted for publication at the
Springer Journal of Intelligent & Robotic Systems, 2020

Table 1 Overview of the paper contributions w.r.t. relevant works in the state of the art. A: capability to drive platforms that can independently control (at least partially) their position and orientation, B: full nonlinear model and control (non-cascaded) of the system dynamics, C: extended/enhanced model of the actuators dynamics including low level constraints, D: controller validated through real experiments with online computation, E: framework suitable to control arbitrarily-designed MRAVs. ✓: implemented, ✗: not implemented.

	[4]	[5]	[6]	[7]	[8]	[9]	[10]	[11]	[12]	[13]	[14]	[15]	[16]	[17]	[18]	[19]	[20]	[21]	[22]	[23]	THIS PAPER
A	✗	✗	✗	✗	✗	✗	✗	✓	✓	✗	✗	✗	✗	✓	✗	✗	✗	✓	✗	✗	✓
B	✗	✗	✗	✗	✗	✗	✗	✓	✗	✓	✓	✗	✗	✓	✗	✗	✗	✓	✗	✗	✓
C	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✓
D	✓	✓	✓	✓	✓	✓	✓	✗	✓	✓	✓	✓	✗	✓	✓	✓	✓	✓	✓	✓	✓
E	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✓

avoid cascaded configuration as well. However, such works either do not include any realistic model of the actuator dynamics and constraints in the control design, or they do not demonstrate the capability of the proposed methods to perform online control of the robot in challenging real experiments. Furthermore, they do not offer a generic framework for the seamless control of both under-actuated and fully-actuated MRAVs with generic designs. On the other hand, the simultaneous accomplishment of all these three objectives constitute the main contribution of this paper.

Another common source of performance limitation is the use of linear/linearized models, see e.g., in [5–10, 12, 23]. Such models have the advantage of typically requiring less computation, in relation to the online resolution of the OCP, but at the detriment of maximum attainable performance. On the other hand, the MPC scheme for local planning and tracking presented in this paper uses a full-order nonlinear model which includes an innovative data-driven description of the actuator dynamics. To effectively limit the increased computational burden, the control algorithm is implemented using a state-of-the-art RTI scheme with partial sensitivity update method, as further explained in the paper.

Therefore, despite the field of MPC-based control for MRAVs is already deeply studied, we believe there is still a considerable margin for interesting research investigation, in particular in relation to the employment of more precise models which take into account more representative constraints for the actuators, can be applied to arbitrarily-designed MRAVs, and are validated through real experiments with online computation, as demonstrated by the novel and unique results in this regard presented in this paper.

To summarize, the contributions of this paper are threefold. First, the take advantage of a novel actuator model that allows to consider as control inputs the derivatives of the forces generated by the multi-rotor vehicle and to leverage the vehicle dynamic capabilities in a better way. Second, the development of a control framework suitable to seamlessly deal with UDT and MDT MRAVs. Third, an extensive and comprehensive validation of the controller by means of real-time simulations and experiments performed with heterogeneous MRAVs, i.e., both with *under-actuated* and *fully-actuated* aerial robots, and both with *fixed* and *orientable* propellers. To the best of our knowledge, this is the first time that a framework with all such relevant characteristics is successfully tested online to control non-specific aerial vehicles with arbitrary propeller arrangements. Following the discussion above, Table 1 provides a summary of the contribution of this paper compared to the main works in the literature.

This paper is structured as follows. First, the mathematical model of a MDT MRAV is described in details, with focus on the novel actuator model development and identification. Then, we describe the MPC implementation details. Finally, we present an extensive and thorough validation campaign, conducted with four heterogeneous robot platforms. The results of both realistic simulations and real experiments for the control of an under-actuated, a fully-actuated, and a convertible MRAV are presented, compared, and discussed. Furthermore, the stabilization of a fully-actuated platform subject to a rotor failure is also targeted. A few summarizing considerations and hints on future work conclude the article.

Notation. In this paper, we denote (column) vectors and matrices in bold font, with lower and upper cases, respectively. The transpose operator is indicated with the superscript $\bullet^\top$. Letter superscripts of vectors represent the reference frame w.r.t which these vectors are expressed. $\mathbf{1}_{i,j}$ denotes the matrix with i rows and j columns with all the elements equal to 1. $\mathbf{A} \otimes \mathbf{B}$ denotes the Kronecker product between the matrices $\mathbf{A}$ and $\mathbf{B}$. For the reader's ease, we collected in Tab. 2 the main symbols related to the modeling used in the paper.

2 Modeling of MRAVs with generic design

2.1 Model of a multi-rotor platform

Multi-rotor platforms are modeled as rigid bodies having mass m, actuated by $n \in \mathbb{N} \setminus \{0\}$ spinning motors cou-

Preprint version,
final version at https://www.springer.com/journal/10846

2.1 Model of a multi-rotor platform

Table 2 Overview of the main symbols used in this paper.

Definition	Symbol
World Inertial Frame	$\mathcal{F}_W$
Multi-rotor Body Frame	$\mathcal{F}_B$
Actuator frame (i-th)	$\mathcal{F}_{A_i}$
Position, velocity, acceleration of O_B in $\mathcal{F}_W$	$\mathbf{p}, \dot{\mathbf{p}}, \ddot{\mathbf{p}}$
Rotation matrix representing $\mathcal{F}_B$ w.r.t. $\mathcal{F}_W$	$\mathbf{R}$
Angular velocity of $\mathcal{F}_B$ w.r.t. $\mathcal{F}_W$, expressed in $\mathcal{F}_B$	$\boldsymbol{\omega}$
Angular acceleration of $\mathcal{F}_B$ w.r.t. $\mathcal{F}_W$, expressed in $\mathcal{F}_B$	$\dot{\boldsymbol{\omega}}$
Position of O_{A_i} in $\mathcal{F}_B$	$\mathbf{p}_{A_i}^B$
Rotation matrix representing $\mathcal{F}_{A_i}$ w.r.t. $\mathcal{F}_B$	$\mathbf{R}_{A_i}^B$
Mass of the vehicle	m
Vehicle's inertia matrix w.r.t. to O_B, expressed in $\mathcal{F}_B$	$\mathbf{J}$
Gravity acceleration	g
Total force acting on the CoM	$\mathbf{f}_B$
Total moment acting on the CoM	$\boldsymbol{\tau}_B$

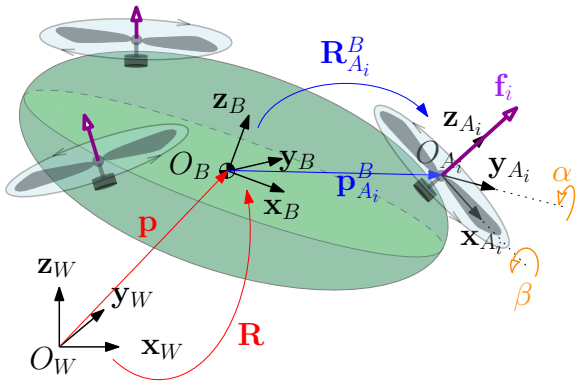

Fig. 1 Schematic representation of a MDT MRAV with its reference frames.

pled with propellers, i.e., $n = 4$ and $n = 6$ in the particular quadrotor and hexarotor models, respectively. Keeping n generic allows to express the model in a non-specific form. With reference to Fig. 1, we denote with $\mathcal{F}_W = O_W$, $\{\mathbf{x}_W, \mathbf{y}_W, \mathbf{z}_W\}$ and $\mathcal{F}_B = O_B$, $\{\mathbf{x}_B, \mathbf{y}_B, \mathbf{z}_B\}$ the world inertial frame and the body frame attached to the MRAV, respectively. The origin of $\mathcal{F}_B$, i.e., O_B, is chosen coincident with the Center of Mass (CoM) of the aerial platform and its position w.r.t. O_W, in $\mathcal{F}_W$, is denoted with $\mathbf{p}_B^W \in \mathbb{R}^3$, shortly indicated with $\mathbf{p}$ in the following. The orientation of $\mathcal{F}_B$ w.r.t. $\mathcal{F}_W$ is represented by the rotation matrix $\mathbf{R}_B^W \in \mathbb{R}^{3\times3}$, denoted with $\mathbf{R}$ for ease of notation. We also define with $\mathcal{F}_{A_i} = O_{A_i}$, $\{\mathbf{x}_{A_i}, \mathbf{y}_{A_i}, \mathbf{z}_{A_i}\}$ the reference frame related to the i-th actuator, $i \in \{1, \dots, n\}$, with O_{A_i} attached to the thrust generation point and $\mathbf{z}_{A_i}$ aligned with the thrust direction. Thanks to this convention, the actuator force expressed in its frame is $\mathbf{f}_i^{A_i} = f_i \mathbf{e}_3$, where $\mathbf{e}_i$, $i=1,2,3$ represents the i-th vector of the canonical basis of $\mathbb{R}^3$. The position of O_{A_i} w.r.t O_B, in $\mathcal{F}_B$, is indicated with $\mathbf{p}_{A_i}^B$, while the orientation of $\mathcal{F}_{A_i}$ w.r.t. $\mathcal{F}_B$ is represented with $\mathbf{R}_{A_i}^B$. The positive definite matrix $\mathbf{J} \in \mathbb{R}^{3\times3}$ denotes the vehicle inertia matrix w.r.t. O_B, expressed in $\mathcal{F}_B$. The angular velocity of $\mathcal{F}_B$ w.r.t. $\mathcal{F}_W$, expressed in $\mathcal{F}_B$, is indicated with $\boldsymbol{\omega}_B^B \in \mathbb{R}^3$ and compactly denoted as $\boldsymbol{\omega}$ in the following. The vehicle orientation kinematics, accounting for the evolution of the rotation matrix $\mathbf{R}$, is described by the well-known equation

$$\dot{\mathbf{R}} = \mathbf{R}\,[\boldsymbol{\omega}]_\times \tag{1}$$

where $[\bullet]_\times \in so(3)$ represents, in general, the skew symmetric matrix associated to any vector $\bullet \in \mathbb{R}^3$.

Using the Newton-Euler formalism, we can derive the dynamics of the aerial platform in order to relate the motion of its CoM, in particular its linear and angular accelerations ($\ddot{\mathbf{p}}$ and $\dot{\boldsymbol{\omega}}$, respectively), to the sum of the forces $\mathbf{f}_B$ and the torques $\boldsymbol{\tau}_B$ acting on this particular point of the rigid body. As traditionally done, we express the translational dynamics in world frame, while keeping the rotational one in body frame. This allows to slightly simplify the form of the equations. Combining them in a compact form, we obtain

$$\begin{bmatrix} m\mathbf{I}_3 & \mathbf{0}_3 \\ \mathbf{0}_3 & \mathbf{J} \end{bmatrix} \begin{bmatrix} \ddot{\mathbf{p}} \\ \dot{\boldsymbol{\omega}} \end{bmatrix} = \begin{bmatrix} -mg\mathbf{e}_3 \\ -\boldsymbol{\omega} \times \mathbf{J}\boldsymbol{\omega} \end{bmatrix} + \begin{bmatrix} \mathbf{R} & \mathbf{0}_3 \\ \mathbf{0}_3 & \mathbf{I}_3 \end{bmatrix} \begin{bmatrix} \mathbf{f}_B^B \\ \boldsymbol{\tau}_B^B \end{bmatrix} \tag{2}$$

where g is the gravitational acceleration and $\mathbf{I}_3 \in \mathbb{R}^{3\times3}$ is the identity matrix of order 3. In order to expand (2), one must explicit the dependence of the body wrench on the forces generated by actuators. The vector $\mathbf{f}_B^B$ is the sum of the actuator forces, properly rotated in body frame, i.e.,

$$\mathbf{f}_B^B = \sum_{i=1}^{n} \mathbf{f}_i^B = \sum_{i=1}^{n} \mathbf{R}_{A_i}^B \mathbf{f}_i^{A_i} = \sum_{i=1}^{n} \mathbf{R}_{A_i}^B \mathbf{e}_3 f_i. \tag{3}$$

On the other hand, the body torque is the result of the moments $\boldsymbol{\tau}_{f_i}$ created by the actuator forces due to their leverage arms and the drag torques $\boldsymbol{\tau}_{d_i}$ which are a byproduct of the counteracting reaction of the air to the propeller rotation.

$$\begin{aligned} \boldsymbol{\tau}_B^B &= \sum_{i=1}^{n} \boldsymbol{\tau}_{f_i}^B + \boldsymbol{\tau}_{d_i}^B = \sum_{i=1}^{n} \mathbf{p}_{A_i}^B \times \mathbf{f}_i^B + c_i c_f^\tau \mathbf{f}_i^B \\ &= \sum_{i=1}^{n} \left([\mathbf{p}_{A_i}^B]_\times + c_i c_f^\tau \mathbf{I}_3\right) \mathbf{R}_{A_i}^B \mathbf{e}_3 f_i. \end{aligned} \tag{4}$$

The constant parameter $c_f^\tau > 0$ is characteristic of the type of propeller and is defined as the intensity ratio between the thrust produced by the propeller rotation and the generated drag torque. Furthermore, c_i is a variable whose value is equal to -1 (respectively, $+1$) in the case the direction of the induced drag torque is opposite (respectively, the same) w.r.t. the generated thrust force, that is the case for a propeller spinning counter-clockwise (respectively, clockwise) w.r.t. its thrust direction. Such coefficient models the fact that the drag torque is always opposed w.r.t. the rotor velocity. In particular, the model used in this paper assumes that the sense of rotation of each rotor is fixed and cannot be reversed. Furthermore, the collective pitch of the propeller blades is modeled as constant. As a consequence, the generated thrust cannot be flipped. Thus, swash-plate designs are

Preprint version,
final version at https://www.springer.com/journal/10846

Accepted for publication at the
Springer Journal of Intelligent & Robotic Systems, 2020

out of the scope of this work. Finally, f_i is the intensity of the produced force, which is related to the controllable spinning rate w_i of motor i by means of the quadratic relation

$$f_i = c_f \mathrm{w}_i^2 \tag{5}$$

where $c_f > 0$ is another propeller-dependent constant parameter to be experimentally identified. Note that (5) is a well-established model in the literature, that has been validated experimentally, e.g., in [33].

We underline that one goal of this paper is to define and guarantee the compliance of the system with meaningful bounds for the actuators, and not to accurately model the physics of the thrust generation. To this purpose, the interest reader is addressed to [34]. Leaving the dependence of the model equations on f_i, see (3)-(4), allows the proposed MPC framework to be seamlessly adaptable to the particular thrust generation model specified by the user. Therefore, also different and more accurate thrust models, such as, e.g., [35] can be easily integrated in our framework.

From (3) and (4), the body wrench can be expressed as a linear combination of the forces produced by the n actuators. Once defined $\boldsymbol{\gamma} = \begin{bmatrix} f_1 \cdots f_n \end{bmatrix}^\top$, we can write

$$\begin{bmatrix} \mathbf{f}_B^B \\ \boldsymbol{\tau}_B^B \end{bmatrix} = \begin{bmatrix} \mathbf{G}_1 \\ \mathbf{G}_2 \end{bmatrix} \boldsymbol{\gamma} = \mathbf{G}\boldsymbol{\gamma} \tag{6}$$

where $\mathbf{G} \in \mathbb{R}^{6\times n}$ is the *allocation matrix*. In particular, its sub-blocks $\mathbf{G}_1$ and $\mathbf{G}_2$ map the actuator forces to the body forces and moments, respectively. Moreover, the j-th column of $\mathbf{G}$, $j \in \{1, \dots, n\}$, refers to the contribution of the j-th actuator force to the total body wrench, being

$$\mathbf{G}(:, j) = \begin{bmatrix} \mathbf{R}_{A_j}^B \mathbf{e}_3 \\ \left([\mathbf{p}_{A_j}^B]_\times + c_j c_f^\tau \mathbf{I}_3 \right) \mathbf{R}_{A_j} \mathbf{e}_3 \end{bmatrix}. \tag{7}$$

The matrix $\mathbf{G}$ maps the vector of actuator force intensities, that belongs to a subset[1] of an n-dimensional space, to body wrenches laying in a subset of a 6-dimensional space. Remark the fact that in the case of a *fully-actuated* MRAV, the allocation matrix has full-rank, while for an *under-actuated* vehicle it has a number of rank deficiencies equal to its under-actuation degree. In the particular case of a UDT platform, we have that $\mathrm{rank}(\mathbf{G}) = 4$, with $\mathrm{rank}(\mathbf{G}_2) = 3$ and $\mathrm{rank}(\mathbf{G}_1) = 1$. This reflects the vehicle capability to exert a body torque in all the directions, disregarding the actuator limits, but a body force along only one direction, i.e., the one of the $\mathbf{z}_B$ axis. A detailed analysis of the allocation matrix rank has been presented in [36], in the particular configuration of a hexarotor with synchronized dual-tilting propellers. The theoretical problem of designing an omni-directional (OD) aerial vehicle, that is a fully-actuated MRAV that can produce any body force inside a spherical shell independently from the body torque, has instead been investigated in [37–39].

The model defined by the equations (2)-(4) describes the dynamics of a MRAV with arbitrarily positioned and rotated actuators. Nevertheless, it contains, like all models, a certain degree of simplification w.r.t. the real system. In the particular case, it neglects the contributions of the gyroscopic effect induced by the conservation of the angular momentum of the propellers, the blade flapping and the rotor induced drag reactions. As far as the gyroscopic effect is concerned, its contribution could be taken into account by adding to the right-side part of the rotational dynamics in (2) a modified version of (3) in [7] that takes into account the fact that the actuators may have different orientations w.r.t. $\mathcal{F}_B$. As one can easily figure out, each term in such equation is scaled with $\mathbf{J}_{A_i}$, that is the inertia tensor of the rotating part of the i-th actuator (composed of the propeller and the rotor). For MRAVs with actuators of small-medium size, that are the ones on which this work focuses its attention, the entries of this matrix are typically 2-3 orders of magnitude smaller than the ones of $\mathbf{J}$. Therefore, the contribution of the gyroscopic effect can be safely neglected in (2). Regarding the blade flapping and the rotor induced drag effects, they are mainly associated with the flexibility and the rigidity of the rotors, respectively [40], and are generated by the interaction of the air with the translating propellers. The results of these aerodynamic effects can be typically observed in UDT platforms as exogenous lateral forces in the x-y plane of the rotors. In the scope of MDT MRAVs, this analysis would be complex to be precisely evaluated and would require to measure the relative speed of the vehicle w.r.t. the wind and to model the possible interactions between the air-flows of different propellers, which is outside the scope of this paper. Moreover, it should be remarked that the behavior of small-medium size rotor-crafts is much more dominated by their thruster characteristics than to aerodynamic forces, cf. [34].

For these reasons, in the line of [13, 19] and many other relevant works, further motivated by the results presented in [33], we decided to neglect the first-order contribution of these two reactions and all other second-order effects arising at very high speed and highly dynamic MRAV maneuvers.

The model developed so far is known in the literature and has been presented for completeness and self-consistency. The true contribution brought by this work regarding the modeling of a MRAV is described in the following paragraph, where we detail a methodology aimed to take into account the dynamics and the limitations of the actuators in a simple and effective way.

[1] Such subset is the Cartesian product of the scalar subsets $\mathbb{F}_i \subset \mathbb{R}^+$ which contain the feasible force values that each actuator can exert.

Preprint version,
final version at https://www.springer.com/journal/10846

Accepted for publication at the
Springer Journal of Intelligent & Robotic Systems, 2020

2.2 State-dependent actuator bounds

In our previous work [31], we already showed the importance of keeping into account the rotor velocity constraints in the MRAV control strategy in order to preserve the system stability. As also claimed in [41], further improvements in the control of MRAVs could be attained by extending the nonlinear model in order to include the motor/blade dynamics and treating the motor voltages as the commanded inputs. However, this would require to accurately model the significant nonlinearities introduced by the *active braking*, to control the system a high rate ($\geq$ 1 KHz) and at low latency ($\leq$ 1 ms), and the availability of further measurements (e.g., the motor currents and spinning velocities).

In [11] a trade-off solution is proposed, considering the rotor accelerations as control input. This strategy allows to put constraints on both the motor velocities and their derivatives. Doing so, the simplistic hypothesis that the spinning velocities of the rotors (and the generated forces, by consequence) can be changed instantaneously, implicitly done by other works in the literature, is abandoned. Constraints on the rotor accelerations are enforced in the OCP resolution, assuming the lower and upper bounds as constant.

However, as corroborated by experimental data, the capability of the rotors to accelerate depends on the motor currents, the blade dynamics, and on other nonlinear effects hidden in the electrical level that could additionally induce an asymmetry between the acceleration and the deceleration constraints, which will in turn indirectly depend on the rotor velocity. For these reasons, we believe that the extended MRAV model should rely on a methodology that can assess the actuators dynamics and constraints in a more accurate way. Since the presence of strong nonlinearities in the closed-loop dynamics of the actuators prevents the use of Bode plots analysis or other linear methods, we propose to derive the model from available data in an alternative way. More specifically, first we experimentally assess how the spinning rate w_i and the acceleration $\dot{\mathrm{w}}_i$ of the rotors, each of which is regulated by an independent embedded Electronic Speed Controller (ESC), should be properly constrained in order to prevent the risk of damaging the motors and to guarantee an accurate force tracking. Secondly, we derive proper constraints for the actuator forces and their derivatives, used by the MPC, in relation to the particular model used to describe the thrust generation. This confers generality to our approach, making it compatible with any other thrust generation model that one wants to adopt.

2.2.1 Experimental assessment of the limitations on the spinning rate w and the acceleration $\dot{w}$ of the rotors

In this paragraph, we present a procedure to experimentally identify appropriate rotor acceleration limits as function of the velocity set-points to the ESCs, allowing to account for the aforementioned nonlinearities in a simple yet effective way. To do this, we use a simple testbed composed of a single Brush-Less Direct-Current BL-DC electric motor that is fixed on a mechanical structure, endowed with a propeller and controlled by a dedicated ESC. The latter is connected to a computer via a serial cable. Using a suitable application, the user should be able to specify the desired rotor velocity w_d, read the measurement w, and measure or estimate the current in input to the motor.

As far as the lower and upper bounds for the rotor velocities are concerned, they can be experimentally identified by producing velocity commands that cause the currents to be at the safety limits, with a certain security margin. This information is available from the motors data-sheets. Such velocity limits should be combined with the ones imposed by the low-level speed controllers, if any.

In order to identify the constraints on the rotor accelerations, i.e., $\underline{\dot{\mathrm{w}}}$ and $\overline{\dot{\mathrm{w}}}$, the actuator should be provided with increasing acceleration commands, centered at different velocity set-points in order to appreciate the dependence of the constraints on the rotor velocity. The profile of the desired rotor velocity trajectory, depicted in Fig. 2, is a sequence of ramps (highlighted with yellow rectangles) centered at given set-points w_h^*, $h \in \mathbb{N} \setminus \{0\}$, that are chosen in order to equally span the feasible set $[\underline{\mathrm{w}}, \overline{\mathrm{w}}]$. The ramp segments are designed with increasing slopes (both positive and negative) over time and separated by rest-intervals where $\dot{\mathrm{w}}_d = 0$, needed to avoid overheating the motor. At this point, the tracking error e_f of the generated force f, mapped from the measured rotor velocity via the thrust generation model, w.r.t. a given desired value f_d can be used as the metric to define the acceleration bounds. Using (5), we have

$$e_f(e_\mathrm{w}, \mathrm{w}_d) = f_d - f = c_f \left(2\mathrm{w}_d e_\mathrm{w} - {e_\mathrm{w}}^2\right) \qquad (8)$$

where $e_\mathrm{w} = \mathrm{w}_d - \mathrm{w}$ is the velocity error. After a standard post processing of the data, mostly consisting in a low-pass filtering of the measured velocity in order to reduce high-frequency noise, by visual inspection of the force error associated with the acceleration intervals centered at each w_h^*, the user can determine the velocity-dependent acceleration limits in relation to the force tracking accuracy (s)he is willing to achieve, i.e., those that guarantee an average force inaccuracy below a chosen threshold ϵ_f. Connecting these values using a linear interpolation, it is possible to have an approximation of $\underline{\dot{\mathrm{w}}}$ and $\overline{\dot{\mathrm{w}}}$ as a function of w.

2.2.2 Definition of the constraints on f and $\dot{f}$

First of all, the values $\underline{\mathrm{w}}$ and $\overline{\mathrm{w}}$ can be translated into the force constraints $\underline{f}$ and $\overline{f}$ by using the force generation model (5). Secondly, once the functions $\underline{\dot{\mathrm{w}}}(\mathrm{w})$ and $\overline{\dot{\mathrm{w}}}(\mathrm{w})$ are available, in order to convert them into constraints on force

Preprint version,
final version at https://www.springer.com/journal/10846

Accepted for publication at the
Springer Journal of Intelligent & Robotic Systems, 2020

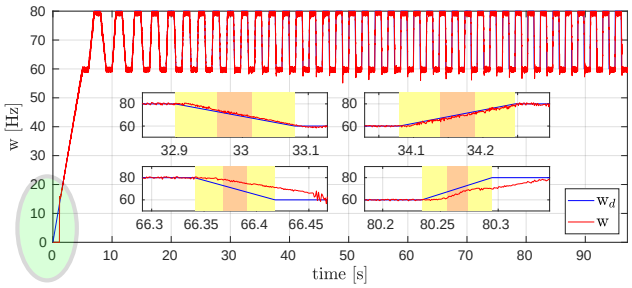

Fig. 2 Trajectory for the identification of the input limits at $\mathrm{w}^* = 70$ Hz (that is the average spinning rotor velocities while the platform hovers) for the hexarotor setup. A series of ramps with increasing slope, which corresponds to growing acceleration commands, is sent to one actuator. The top and bottom sub-plots outline intervals where the tracking of the velocity command is good and bad, respectively. In particular, remark on the green ellipse that once the motor is activated, it has a minimum spinning velocity $\underline{\mathrm{w}} = 16$ Hz below which it can't physically rotate. This has to be kept into account by the MPC.

derivatives, one can easily compute the time-derivative of (5), obtaining

$$\dot{f} = \frac{\partial f}{\partial \mathrm{w}} \frac{\partial \mathrm{w}}{\partial t} = 2 c_f \mathrm{w} \dot{\mathrm{w}}. \tag{9}$$

The expression of the state dependent input constraints $\overline{\dot{f}}(f)$ and $\underline{\dot{f}}(f)$ are finally obtained from (5) and (9). We stress the fact that another model for the thrust generation might be used. In that case (5), and consequently (8) and (9), should be changed according to the new thrust model.

As a final remark, it should be noted that the proposed procedure does not require a force/torque sensor.

2.2.3 Application of the identification procedure to hardware setup

In the following, we describe how we concretely apply the previously-described procedure to two different hardware setups. The first one, shortly named *setup I*, is composed of a MikroKopter[2] electric motor MK3638 coupled with a 12X4.5' propeller and controlled by a BL-Ctrl V2.0 ESC. The low-level control of the rotor velocity is performed in closed-loop employing the Adaptive Bias Adaptive Gain (ABAG) algorithm, whose details can be found in [42].

In this setup, being the one of our (custom-made) fully-actuated hexarotors, the constraints on the minimum and maximum velocities are related to the properties of the *closed-loop* rotor velocity controller. Specifically, the actual rotor velocity is estimated by the low-level controller without any additional sensor and the quality of such estimation is proportional to the rotor speed. This causes the velocity to have a lower bound, in order to be properly estimated by the controller with a certain precision. On the other hand, the limited arithmetic capabilities of the ESC micro-controller (which allows only 8-bit additions and has no floating point unit) translates into a velocity upper bound, cf. [42]. In this case, we identified $\underline{\mathrm{w}} = 16\,\mathrm{Hz}$ and $\overline{\mathrm{w}} = 102\,\mathrm{Hz}$. In particular, the upper limit satisfies the maximum current limitation of 20 A reported in the motor data-sheet. Finally, using (5) we obtained the limits $\underline{f} \approx 0.25\,\mathrm{N}$ and $\overline{f} \approx 10.3\,\mathrm{N}$ used to constrain the OCP resolution in the MPC algorithm.

As far as the identification of the acceleration limits is concerned, we generated a set of increasing $\dot{\mathrm{w}}$ spanning the range $\pm[20, 300]\,\mathrm{Hz/s}$ with a step of $10\,\mathrm{Hz/s}$, centered at a given average velocity level w_h^*. Each ramp fragment takes values in the set $[\mathrm{w}_h^* - \delta_h, \mathrm{w}_h^* + \delta_h]$, with $\delta_h = 10\mathrm{Hz}$. With reference to Fig. 2, for each ramp we select the 30% of the total samples which are centered in the middle of the interval (highlighted with orange rectangles in Fig. 2) and compute the correspondent force error using (8). The operation is repeated at different set-points w_h^* in the set $[30, 90]$ Hz with a step of 10 Hz, in order to span the set of admissible velocities previously estimated. The plots of the force error trends related to *setup I* are shown in Fig. 4. In each subplot, notice that the number of samples related to increasing values of $|\dot{\mathrm{w}}|$ is gradually decreasing. This happens because an increase in the ramp slopes is associated with a decrease in the time duration associated with the segments. Remark three facts: (i) At the same velocity set-points, increasing force errors are associated with increasing acceleration values, on average. This suggests that high acceleration references (of both signs) are difficult to be tracked and fosters the idea to constrain them with lower and upper bounds. (ii) For different set-point velocities, the profile of the force error at corresponding acceleration intervals is different. This confirms the claim that the limits are velocity-dependent. In particular, we observe that while increasing values of set-points seem to cause increasing force error for positive accelerations, such trend is not pursued by negative accelerations. A reasonable explanation for such effect could be the fact that the active braking, which intervenes only for negative accelerations, is not behaving in the same way for different velocity levels. (iii) At the same velocity set-points, the force error e_f associated with negative accelerations is larger, on average, w.r.t. the one associated with positive accelerations. This reveals that, despite the use of the active-braking, the deceleration of a rotor produces a worse force tracking than the corresponding acceleration.

In order to identify the acceleration limits $\underline{\dot{\mathrm{w}}}$ and $\overline{\dot{\mathrm{w}}}$, we defined $\epsilon_f \approx 0.2\,\mathrm{N}$ as the force error threshold, admitting

Table 3 Identified acceleration limits for *setup I*.

w [Hz]	30	40	50	60	70	80	90
$\underline{\dot{\mathrm{w}}}$ [Hz/s]	−120	−160	−200	−140	−160	−160	−140
$\overline{\dot{\mathrm{w}}}$ [Hz/s]	200	200	200	160	180	180	180

[2] http://www.mikrokopter.de/en/home

Preprint version,
final version at https://www.springer.com/journal/10846

2.3 State-space model for discrete-time control

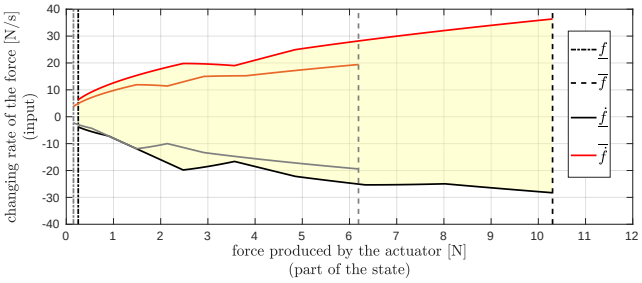

Fig. 3 State and input constraints given to the NMPC in relation to the two hardware setups used in the experiments. Darker and lighter colored lines are referred to *setup I* and *setup II*, respectively.

slightly bigger values ($\approx 0.3\,\mathrm{N}$) at high velocity set-points. As we will see in the experimental validation plots, such value generates conservative limits that preserve the platform stability also during agile trajectory tracking. As a general rule, such threshold value shall depend on the particular robot task. The identified acceleration limits related to *setup I* are collected in Tab. 3, where velocity data are expressed in Hz, while acceleration ones in Hz/s. Interpolating these values with linear functions and using (5) and (9), allowed us to obtain the force derivative constraints as function of the instantaneous thrust forces.

A second hardware setup (*setup II*) is analyzed, i.e., that one of the available under-actuated quadrotor, which combines a MK2832/35 motor with a 10X4.5' propeller from MikroKopter, controlled by the same ESC and closed-loop algorithm of *setup I*. The profile of the constraints for the actuator forces and their derivatives, related to the two setups, are depicted with different colors in the plot of Fig. 3, where the admissible set of values for both cases are represented with the yellow area. Consistently with the previous results, the limits on positive and negative thrust derivatives are not perfectly symmetric.

2.3 State-space model for discrete-time control

Let us define the state vector $\mathbf{x}$ and the input vector $\mathbf{u}$ as

$$\mathbf{x} := \left[\mathbf{p}^\top\ \dot{\mathbf{p}}^\top\ \boldsymbol{\eta}^\top\ \boldsymbol{\omega}^\top\ \boldsymbol{\gamma}^\top\right]^\top \tag{10}$$

$$\mathbf{u} := \dot{\boldsymbol{\gamma}} \tag{11}$$

with $\boldsymbol{\eta} \in \mathbb{R}^{n_\eta}$ being the vector used for concisely representing the platform orientation. Specifically, for the experimental validation we chose a minimum representation with three angles (see the section related to the experimental validation for a detailed discussion about pros and cons of minimal representations). With reference to (10), it is worth to remark the fact that the actuator forces $\boldsymbol{\gamma}$, which in other works were either discarded from the model or assumed as the input, are considered here as part of the state. In particular, all the quantities which compose the state are assumed to be measurable (cf. Sec. 4.1 for a discussion of the employed sensors). In view of this, the control scheme will be implemented without resorting to a dedicated state-observer.

The expression of the map $\mathbf{f}(\bullet)$ relating $\dot{\mathbf{x}}$ to $\mathbf{x}$ and $\mathbf{u}$, i.e.,

$$\dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t), \mathbf{u}(t)), \tag{12}$$

can be obtained from (1)-(4), according to the previous definition of $\mathbf{x}$ and $\mathbf{u}$. For digital control purposes, the continuous-time model in (12) is discretized (in the particualr case using a fixed step 4^{th} order explicit Runge-Kutta integrator) yielding the following discrete-time model

$$\mathbf{x}_{k+1} = \boldsymbol{\phi}(\mathbf{x}_k, \mathbf{u}_k), \quad k = 0, 1, \ldots, N-1 \tag{13}$$

where, for ease of notation, $\mathbf{x}_k = \mathbf{x}(kT)$ being T the MPC sampling time and $\mathbf{u}(t) = \mathbf{u}_k$ for $t \in [kT, (k+1)T)$.

3 NMPC for MRAVs with generic design

The goal of this section is to devise an MPC controller able to simultaneously address the problem of local reference trajectory planning and that of stabilizing the vehicle dynamics. Specifically, we aim to track a reference trajectory denoted $(\mathbf{p}_r(t), \boldsymbol{\eta}_r(t))$ given by a generic global planner. We assume $(\mathbf{p}_r(t), \boldsymbol{\eta}_r(t))$ to be twice continuously differentiable. In order to guarantee smoothness properties of the generated trajectory, we force our algorithm to be able to drive also the derivatives of the state variables toward the corresponding ones of the reference trajectory. Therefore, we introduce the following enlarged reference signal

$$\mathbf{y}_r(t) = \left[\mathbf{p}_r^\top(t)\ \dot{\mathbf{p}}_r^\top(t)\ \ddot{\mathbf{p}}_r^\top(t)\ \boldsymbol{\eta}_r^\top(t)\ \boldsymbol{\omega}_r^\top(t)\ \dot{\boldsymbol{\omega}}_r^\top(t)\right]^\top \tag{14}$$

and, accordingly, we define the output map as

$$\mathbf{y}(t) = \mathbf{h}\left(\mathbf{x}(t), \mathbf{u}(t)\right) = \begin{bmatrix} \mathbf{p}(t) \\ \dot{\mathbf{p}}(t) \\ \ddot{\mathbf{p}}\left(\mathbf{x}(t), \mathbf{u}(t)\right) \\ \boldsymbol{\eta}(t) \\ \boldsymbol{\omega}(t) \\ \dot{\boldsymbol{\omega}}\left(\mathbf{x}(t), \mathbf{u}(t)\right) \end{bmatrix}. \tag{15}$$

For clarity observe that $\mathbf{p}(t), \dot{\mathbf{p}}(t), \boldsymbol{\eta}(t), \boldsymbol{\omega}(t)$ are measured sub-vectors of the state, while $\ddot{\mathbf{p}}, \dot{\boldsymbol{\omega}}$ are functions of $\mathbf{x}(t), \mathbf{u}(t)$, and, in particular, sub-components of the map $\mathbf{f}$ in (12).

Finally, we define $\mathbf{y}_{r,k}$, $\mathbf{y}_k$ as the discretized version of $\mathbf{y}_r(t)$ and $\mathbf{y}(t)$, respectively, i.e., $\mathbf{y}_{r,k} = \mathbf{y}_r(kT)$, $\mathbf{y}_k = \mathbf{y}(kT)$. The OCP to be solved at time kT, given the current state $\mathbf{x}_k$, is formulated as

Accepted for publication at the
Springer Journal of Intelligent & Robotic Systems, 2020

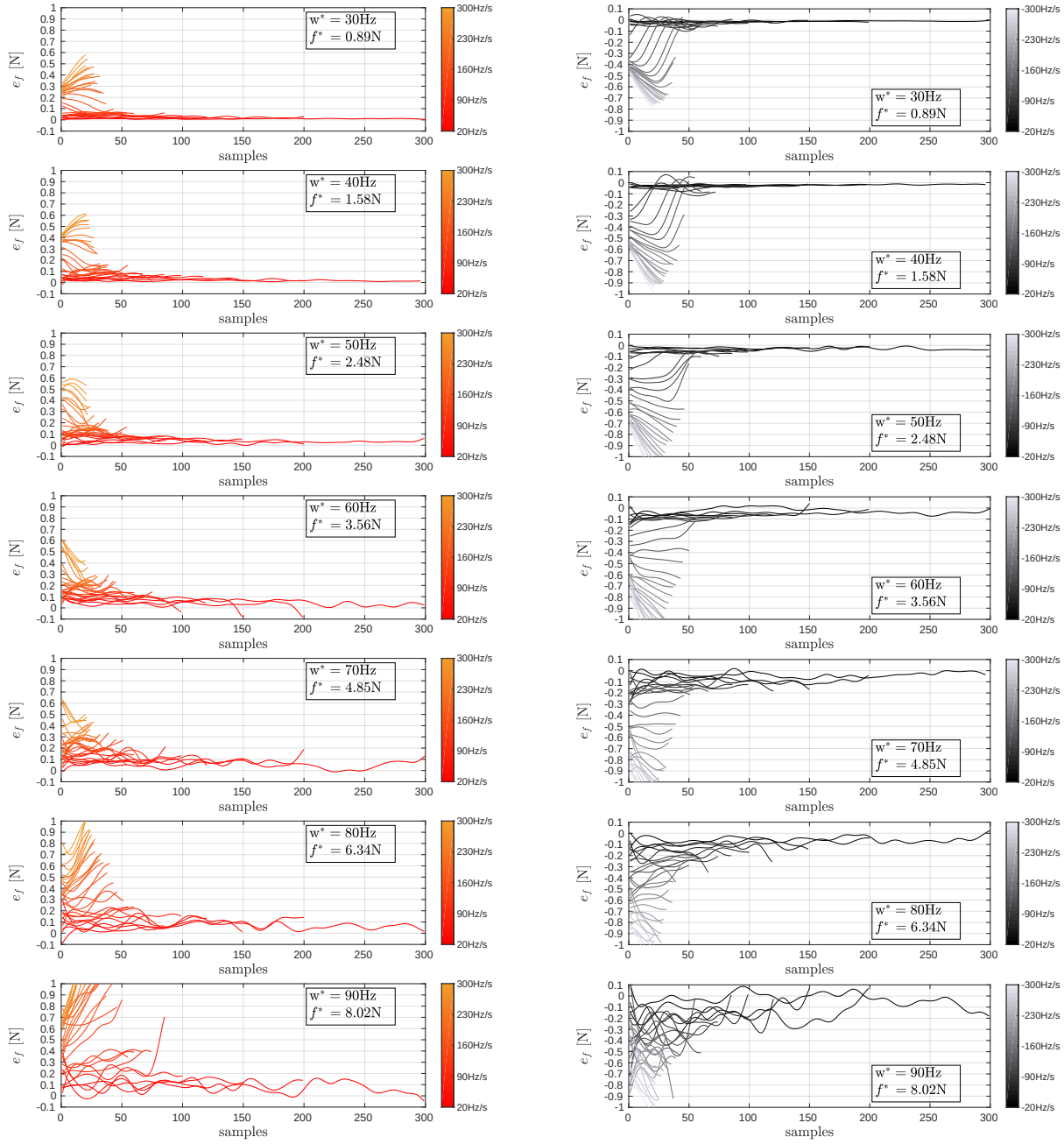

Fig. 4 Plots of the force error trends, associated to the acceleration intervals at different set-point velocities w_h^*, related to *setup I*. Positive and negative acc. are depicted on the left and right column, respectively. The acceleration-dependent errors are represented with different color shades.

$$\min_{\substack{\hat{\mathbf{x}}_0,\dots,\hat{\mathbf{x}}_N \\ \hat{\mathbf{u}}_0,\dots,\hat{\mathbf{u}}_{N-1}}} \sum_{h=0}^{N-1} \left\{ \|\hat{\mathbf{y}}_h - \mathbf{y}_{r,k+h}\|^2_{\mathbf{Q}_h} + \|\hat{\mathbf{u}}_h\|^2_{\mathbf{R}_h} \right\} + \\ + \|\hat{\mathbf{y}}_N - \mathbf{y}_{r,k+N}\|^2_{\mathbf{Q}_N} \tag{16}$$

$$\text{s.t.} \quad \hat{\mathbf{x}}_0 = \mathbf{x}_k \tag{17}$$

$$\hat{\mathbf{x}}_{h+1} = \boldsymbol{\phi}(\hat{\mathbf{x}}_h, \hat{\mathbf{u}}_h), \quad h=0,1,\dots,N-1, \tag{18}$$

$$\hat{\mathbf{y}}_h = \mathbf{h}(\hat{\mathbf{x}}_h, \hat{\mathbf{u}}_h), \quad h=0,1,\dots,N, \tag{19}$$

$$\underline{\boldsymbol{\gamma}} \leq \mathbf{M}\hat{\mathbf{x}}_h \leq \overline{\boldsymbol{\gamma}}, \quad h=0,1,\dots,N, \tag{20}$$

$$\underline{\dot{\boldsymbol{\gamma}}}_{k+h} \leq \hat{\mathbf{u}}_h \leq \overline{\dot{\boldsymbol{\gamma}}}_{k+h}, \quad h=0,1,\dots,N-1, \tag{21}$$

where $\mathbf{Q}_h, \mathbf{R}_h$ are semidefinite positive matrices and matrix $\mathbf{M}$ is defined in order to select only the n elements of the state $\mathbf{x}$ corresponding to the actuator forces, that is,

$$\mathbf{M} = \begin{bmatrix} \mathbf{0}_{n\times(9+n_\eta)} & \mathbf{I}_n \end{bmatrix}. \tag{22}$$

The bounds $\overline{\boldsymbol{\gamma}}, \underline{\boldsymbol{\gamma}}$, depend on the quantities $\overline{f}$, $\underline{f}$ characterized in the previous section and, compactly, they are defined as

$$\overline{\boldsymbol{\gamma}} = \mathbf{1}_{6\times 1} \otimes \overline{f} \tag{23}$$

$$\underline{\boldsymbol{\gamma}} = \mathbf{1}_{6\times 1} \otimes \underline{f}. \tag{24}$$

Preprint version,
final version at https://www.springer.com/journal/10846

Furthermore, the bounds $\underline{\dot{\gamma}}_{k+h}, \overline{\dot{\gamma}}_{k+h}$, $h = 0, 1, \ldots, N-1$, depend on the time-varying and state dependent quantities $\overline{\dot{f}}(f)$, $\underline{\dot{f}}(f)$ and, precisely, they should be defined as

$$\overline{\dot{\gamma}}_{k+h} = \left[\overline{\dot{f}}(f_{1,k+h}), \ldots, \overline{\dot{f}}(f_{n,k+h})\right]^\top \tag{25}$$

$$\underline{\dot{\gamma}}_{k+h} = \left[\underline{\dot{f}}(f_{1,k+h}), \ldots, \underline{\dot{f}}(f_{n,k+h})\right]^\top \tag{26}$$

Observe that constraints in (21), with $\overline{\dot{\gamma}}_{k+h}$ and $\underline{\dot{\gamma}}_{k+h}$ defined as above are highly non-linear. In order to retain linearity of these constraints in the OCP, we consider the following alternative definition for $\overline{\dot{\gamma}}_{k+h}$ and $\underline{\dot{\gamma}}_{k+h}$

$$\overline{\dot{\gamma}}_{k+h} = \left[\overline{\dot{f}}(\tilde{f}_{1,k+h}), \ldots, \overline{\dot{f}}(\tilde{f}_{n,k+h})\right]^\top \tag{27}$$

$$\underline{\dot{\gamma}}_{k+h} = \left[\underline{\dot{f}}(\tilde{f}_{1,k+h}), \ldots, \underline{\dot{f}}(\tilde{f}_{n,k+h})\right]^\top \tag{28}$$

where $\tilde{f}_{i,k+h}$ are independent of the decision variables of the OCP at time $t = kT$. Different choices can be taken, for example keeping the constraints constant along the horizon

$$\tilde{f}_{i,k+h} = f_{i,k}, \quad h = 0, \ldots, N-1.$$

Alternatively, $\tilde{f}_{i,k+h}$ can be selected in a time-varying fashion based on the solution of the previous OCP obtained at instant $t = (k-1)T$.

The solution to the OCP, at a given time step k consists of the optimal values $\hat{\mathbf{x}}_{0|k}, \ldots, \hat{\mathbf{x}}_{N|k}, \hat{\mathbf{u}}_{0|k}, \ldots, \hat{\mathbf{u}}_{N-1|k}$. According to the receding horizon principle [43], the input value $\mathbf{u}_k = \hat{\mathbf{u}}_{0|k}$ is applied, and the procedure is repeated at the subsequent time step $k+1$.

Some remarks are due at this point, concerning the problem formulation. Regarding the stability-related properties of our scheme, first of all note that the problem addressed here consists of tracking a trajectory generated, possibly without any regard of the vehicle model, by a generic global planner. In particular, we avoid on purpose any feasibility assumptions of the reference trajectory w.r.t. the robot, in order to test the NMPC framework capability to locally re-generate and track a trajectory which is compatible with the system dynamics and with the actuator constraints. This motivates the claim that the proposed algorithm can seamlessly deal with arbitrarily-designed MRAVs, without the need for a preliminary analysis on the system dynamics. For example, if the system is under-actuated and differentially flat, our framework will automatically recognize such property and exploit it, thus considerably simplifying the reference trajectory generation problem.

Under this general assumption, stability (in a strict sense) of the reference trajectory cannot be guaranteed, since the guarantee to track the given set-point is ensured only provided that the trajectory is generated compatibly with the system dynamics and the actuator constraints. For particular implementations of nonlinear MPC, in case of feasible set-points or reference trajectories, the stability of the closed-loop system can be conferred, for example, by designing particular terminal constraints and appropriate terminal penalties on the state. A compelling work reviewing the main essential principles that ensures stability has been presented in the survey [43]. However, the difficulty in explicitly computing the terminal set and the terminal cost function for general nonlinear systems remains quite dissuasive in real-life applications [44].

On the other hand, it has been shown that under the assumption that the reference trajectory is consistent with the vehicle dynamics (e.g., as in [45]), stability guarantees could be provided by selecting a sufficiently large prediction horizon length N relying on [46, 47]. Such approach has been applied for path following in a robotic scenario, e.g., in [48]. In line with many other works dealing with NMPC applied to aerial vehicles, we preferred to heuristically follow this methodology. To determine the minimum length of the horizon, which directly affects the dimension of the optimization problem, we have performed preliminary simulations with different trajectories, robot models, and prediction horizon lengths, carrying out a trade-off between stability performance and computational burden. Considering that a formal proof of the controller stability is out of the scope of this paper, we leave the analytic study of the optimal prediction horizon length as well as a formal discussion of the closed-loop system stability for future work. Practically, throughout all the experimental tests, the NMPC algorithm was always able to stabilize the robot along a re-computed trajectory, even in case the reference one is (on purpose) not feasible with respect to the robot dynamics and actuator constraints.

In relation to the stability problem, it is worth to briefly discuss the controllability and observability of the systems to be controlled. Regarding the former property, which is related to the capability of the input to affect the evolution of the system state, we leveraged previous theoretical results to assess the existence of feasible input trajectories capable to steer the state of the system towards the tested reference trajectories. Whenever the specific motion was not feasible for the particular dynamic constraints of the platform at hand, we allowed the local re-generation of a feasible trajectory. In this way, we ensure the system controllability to "the closest" feasible trajectory in relation to the original reference. In particular, while fully-actuated systems can track a full-pose decoupled trajectory provided that the actuator constraints are not violated [31], we know from previous theoretical results that the trajectory of under-actuated vehicles has to satisfy the flatness-property [49]. Moreover, results on the reduced controllability of particular fully-actuated MRAVs after a propeller failure are available from [50, 51]. Ultimately, we also tested the controllability of all the pre-

Preprint version,
final version at https://www.springer.com/journal/10846

Accepted for publication at the
Springer Journal of Intelligent & Robotic Systems, 2020

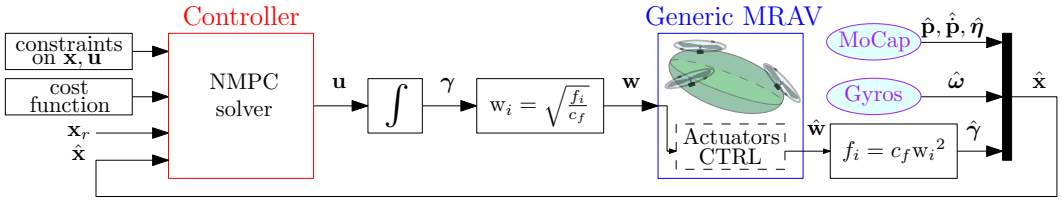

Fig. 5 Block diagram of the experimental setup architecture. The main components are highlighted with different colors.

sented MRAV systems in relation to the target trajectories in a preliminary phase of extensive simulations.

As far as the observability is concerned, which describes the possibility of inferring the internal state of a system from knowledge of its external outputs, we do not have any concern in this sense as the full state of the aerial vehicle, cf. (10), is assumed measurable from the available sensors. Considering that we do not make use of any state estimator in this work, a formal analysis of the system observability is not needed, in this case. Details on the design of a fault-tolerance NMPC scheme for systems with sensor faults, which falls outside the scope of this paper, can be found in [52].

Another important feature of MPC-based algorithms is recursive feasibility, i.e., the guarantee that the OCP always admits a solution. In our practical implementation we have adopted the widespread solution of guaranteeing it by enforcing the (slightly tightened) constraints in a soft way using slack variables. Practically, alongside our extensive experimental campaign, the algorithm was always able to find a solution.

To conclude the extensive presentation of the proposed NMC scheme, the implementation details related to the resolution of the OCP are discussed in the following.

3.1 Implementation details for the OCP resolution

The control algorithm is implemented using the state-of-the-art Real Time Iteration (RTI) scheme, see [53], embedding the multiple shooting method, cf. [54]. The RTI scheme performs a single sequential quadratic Programming (SQP) iteration to solve the OCP. To do this, a linearization of the system constraints (18) and (19) is performed to obtain a quadratic programming (QP) problem, to be solved at each sampling time. To reduce the computational time, in [55] a procedure called partial sensitivity update is proposed, where the constraint linearization is updated only if the dynamics around the generated trajectory exhibits a certain degree of nonlinearity. To reduce the computational complexity, the QP problem is condensed using the algorithms discussed in [56]. The required linear algebra routines are implemented using OpenBLAS [3]. The resulting dense QP is solved by qpOASES, see [57], which employs on-line active-set method with warm-start strategy.

[3] https://github.com/xianyi/OpenBLAS

According to the common practice, the sampling time must be selected to be as small as possible, to make the control system sufficiently reactive. On the other hand, it must also be sufficiently larger than the average computational time. However note that, despite this, there is no guarantee that a solution to the OCP is always available in due time at each time step. If, at a given instant (say at step k), the time required to compute the solution is occasionally larger than a given threshold, a back-up solution must be taken to guarantee reliability of the control system. In this paper, this solution consists of taking the possibly sub-optimal but admissible value $\mathbf{u}_k = \hat{\mathbf{u}}_{1|k-1}$, computed as part of the solution to the OCP at time $k-1$.

4 Experimental validation

In this section we show and thoroughly discuss the experimental results obtained from the application of the proposed NMPC algorithm to the aerial robot prototypes built, and in some cases conceived, in the laboratory facility of LAAS-CNRS - the interested reader is as well referred to the attached multimedia file. First of all, we present the experimental setup, with a focus on the description of both the hardware and the software components, and on the implementation details of the NMPC strategy. Then, we analyze the outcomes of the experiments achieved with two different kinds of MRAVs, i.e., an under-actuated UDT quadrotor and a MDT (in particular, also fully-actuated) hexarotor with tilted propellers. The goal of this investigation is to demonstrate the precision of our approach and, above all, its potential applicability to any arbitrarily-designed MRAV. The robots are required to perform tracking experiments: the reference trajectories are designed both to test the re-generation capability of the algorithm, to highlight the different behaviors of UDT and MDT platforms, and to assess the solution compliance with the constraints previously identified in the case of fast maneuvers.

In order to better appreciate the results achieved in the experimental validation with the proposed NMPC framework, we point the reader to the attached video.

Preprint version,
final version at https://www.springer.com/journal/10846

4.1 Experimental setup

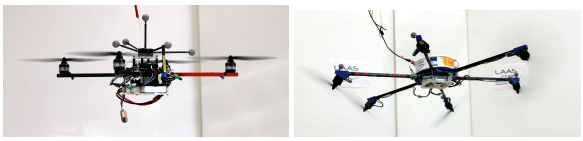

Fig. 6 Photos of the quadrotor (left) and the Tilt-Hex (right).

4.1 Experimental setup

The experimental setup architecture, whose block diagram is portrayed in Fig. 5, can be conceptually divided into three main components: the NMPC controller, which periodically computes the input of the actuator controllers (the ESCs), the physical aerial robot to be controlled, and the sensors, used to retrieve the information about the MRAV state that is employed as feedback in the closed-loop control strategy. Each block exchanges information with the others thanks to a properly-designed software architecture.

The predictive controller is implemented using MATMPC, a recently-developed MATLAB-based nonlinear MPC toolbox, see [58]. Its algorithmic routines are written using MATLAB C API and available as MEX functions. The tool supports fixed step Runge-Kutta (RK) integrator for multiple shooting and obtains the derivatives that are needed to perform the optimization from the toolbox CasADi[4]. The OCP described by (16)-(21) is solved by the external solver qpOASES[5] by [57], integrating the RTI algorithm. Such an implementation has been chosen mainly due to the particular ease of test and development of MATLAB/Simulink® compared to pure C/C++. The presented NMPC algorithm is executed on a ground-station PC equipped with an Intel® 2.60GHz Core™ i7-6700HQ CPU (x8) and 32 GB RAM which runs the Linux Ubuntu 16.04 LTS operating system. As it can be observed from Fig. 5, the control input $\mathbf{u}$, which provides the actuators' force derivatives references, is integrated and then converted into a rotor velocity command $\mathbf{w}$, thanks to the inversion of the force generation model. The resulting velocity set-points are finally transferred to the module of the low-level controllers on-board the aerial platform, by means of a serial cable.

As far as the aerial robots are concerned, we tested our control algorithm with the two heterogeneous MRAVs depicted in Fig. 6. The first one, shown on the left, is an underactuated UDT platform, a quadrotor, with four collinear rotors. Apart from some custom-made features realized in-house with 3D printed components, like the battery support, most of the platform is built by assembling off-the-shelf parts from MikroKopter. The second robot, illustrated on the right of the Fig. 6, is a fully-actuated MDT platform with six non-collinear tilted rotors, from which it inherits the name 'Tilt-Hex'. In this case, the prototype has been completely designed and manufactured in our laboratory, and has already been presented in some of our previous contributions [2, 31]. The values of the physical parameters used in the MRAVs models are summarized in Tabs. 4 and 5. In particular, α and β are defined as the actuator rotation angles around $\mathbf{x}_{A_i}$ and $\mathbf{y}_{A_i}$, respectively, as shown in Fig. 1.

Table 4 Physical parameters of the quadrotor.

Quadrotor		
Parameter	Value	Unit
m	1.042	Kg
$\mathbf{J}(:,1)$	$[0.015\ 0\ 0]^\top$	Kg m^2
$\mathbf{J}(:,2)$	$[0\ 0.015\ 0]^\top$	Kg m^2
$\mathbf{J}(:,3)$	$[0\ 0\ 0.015]^\top$	Kg m^2
c_i	$(-1)^{i-1}$	[ ]
c_f^τ	1.69e−2	m
c_f	5.95e−4	N/Hz^2
$\mathbf{R}_{A_i}^B$	$\mathbf{R}_z((i-1)\frac{\pi}{2}))\mathbf{R}_x(\alpha)\mathbf{R}_y(\beta)$	[ ]
$\mathbf{p}_{A_i}^B$	$\mathbf{R}_z((i-1)\frac{\pi}{2}))[l\ 0\ 0]^\top$	[ ]
α	0	deg
β	0	deg
l	0.23	m

The main sensors integrated in our experimental framework are the onboard gyroscope, the Motion Capture system, and speedometers of each propeller rotational speed:

- the Gyroscope measures the rotational velocity of the vehicle around each of the body frame axis;
- the Motion Capture (MoCap) system provides the information regarding the robot position and orientation w.r.t. the inertial reference frame, whose origin is fixed in a particular point of the robots workspace. The platform linear velocity is numerically computed online from the position measurements, using multi-sample least squares model fitting;
- the rotor spinning velocities are measured by the low-level ESC controller by computing the time elapsed between two phase switches (which depends on the motor number of poles) and reducing the measurement noise with an exponential moving average filter. Ultimately, the rotor velocities are converted into the actuator forces, thanks to the force generation model, and used to complete the information of the measured full-state $\hat{\mathbf{x}}$ of the MRAV.

Note that the accelerometers have been disregarded from the sensor fusion since we assessed that the noise in their measurements was causing an offset in the estimation of the linear velocity, which motivated the numerical computation of the latter. In general, the effect of such velocity offset on the tracking performance is quite more evident on predictive controllers w.r.t. reactive ones, given the fact that a wrong state estimation generates an erroneous evolution of the model internally simulated and, in turn, a misleading

[4] https://github.com/casadi/casadi/wiki
[5] https://projects.coin-or.org/qpOASES/wiki

Accepted for publication at the
Springer Journal of Intelligent & Robotic Systems, 2020

Table 5 Physical parameters of the Tilt-Hex.

Tilt-Hex		
Parameter	Value	Unit
m	1.86	Kg
$\mathbf{J}(:,1)$	$[0.11\ 0\ 0]^\top$	Kg m^2
$\mathbf{J}(:,2)$	$[0\ 0.11\ 0]^\top$	Kg m^2
$\mathbf{J}(:,3)$	$[0\ 0\ 0.19]^\top$	Kg m^2
c_i	$(-1)^{i-1}$	[ ]
c_f^τ	1.9e−2	m
c_f	9.9e−4	N/Hz^2
$\mathbf{R}_{A_i}^B$	$\mathbf{R}_z\left((i-1)\frac{\pi}{3}\right)\mathbf{R}_x(\alpha_i)\mathbf{R}_y(\beta)$	[ ]
$\mathbf{p}_{A_i}^B$	$\mathbf{R}_z\left((i-1)\frac{\pi}{3}\right)[\ell\ 0\ 0]^\top$	[ ]
α_i	$(-1)^i\ 35$	deg
β	-25	deg
ℓ	0.368	m

control input that finally produces an inaccurate trajectory tracking.

In order to design the software architecture, we rely on the GenoM3[6] abstraction level, which allows to encapsulate software functions inside independent components. More in detail, it is used as a wrapper for the robot low-level controller and the sensors. This allows to obtain high flexibility in the development and in the use of the components. With reference to our architecture, the software in MATLAB /Simulink® communicates with the GenoM3 modules using the Robot Operating System (ROS) middleware, which is compliant with the soft real-time constraints required for our experiments, i.e., a control bandwidth larger than 200Hz and a latency smaller than 10ms. Since MATLAB/Simulink® is not meant for a hard real-time execution, the hardware is commanded via the GenoM3 components, which essentially behave like drivers.

4.1.1 Implementation details

For all the experiments presented in this paper, we chose a prediction horizon of $t_H = 1$s, sampled at $N+1 = 11$ shooting points. Therefore, the discretization time of the nonlinear MPC algorithm, being the length of one of the N intervals, results $T = 0.1$s. Even though the internal MPC prediction is performed at 10Hz, the controller runs at a frequency always larger than 200Hz. Such technique, employed by many state-of-the-art contributions, e.g., [14], allows the predictive algorithm to simulate the model along a wider prediction horizon with less computational effort. Indeed, as observed by [59], the number of discretization nodes roughly increases the computational time t_{solv} by $O(N^2)$. Basically, one should guarantees a control sample time T_{ctrl} at least equal to time t_{solv} needed for the algorithm to solve the OCP. On the other hand, the prediction horizon should be long enough to cover at least the time of one controller iteration. In mathematical terms, this translates in the following chain of inequalities

$$t_{\text{solv}} \leq T_{\text{ctrl}} \leq T \leq t_H \tag{29}$$

As far as the representation of the robot orientation is concerned, for the particular experiments presented in this paper we decided to use a minimal parametrization with three angles, in particular the $3-2-1$ one (yaw-pitch-roll), i.e.,

$$\boldsymbol{\eta} = \begin{bmatrix}\phi\ \theta\ \psi\end{bmatrix}^\top \tag{30}$$

With reference to this ordered sequence, we have that

$$\begin{aligned}\mathbf{R} &= \mathbf{R}_\mathbf{z}(\psi)\mathbf{R}_\mathbf{y}(\theta)\mathbf{R}_\mathbf{x}(\phi)\\ &= \begin{bmatrix} c_\theta c_\psi & s_\phi s_\theta c_\psi - c_\phi s_\psi & s_\phi s_\psi + c_\phi s_\theta c_\psi \\ c_\theta s_\psi & c_\phi c_\psi + s_\phi s_\theta s_\psi & c_\phi s_\theta s_\psi - s_\phi c_\psi \\ -s_\theta & s_\phi c_\theta & c_\phi c_\theta \end{bmatrix}\end{aligned} \tag{31}$$

where $\mathbf{R}_\bullet(\alpha)$ denotes a rotation around one of the main body frame axes $\{\mathbf{x}, \mathbf{y}, \mathbf{z}\}_B$ of an angle α, while s_α, c_α indicate $\sin(\alpha)$ and $\cos(\alpha)$, respectively. Using this convention, we can express the body frame angular velocity as a function of the vector $\dot{\boldsymbol{\eta}}$, that contains the so-called *Euler rates*

$$\boldsymbol{\omega} = \mathbf{T}\dot{\boldsymbol{\eta}} \tag{32}$$

In particular, with reference to the specific parametrization of (31), we have

$$\mathbf{T} = \begin{bmatrix} 1 & 0 & -s_\theta \\ 0 & c_\phi & s_\phi c_\theta \\ 0 & -s_\phi & c_\phi c_\theta \end{bmatrix}. \tag{33}$$

Inverting (32) allows to write explicitly the Euler rates as a function of the body angular velocity (expressed in body frame) in the NMPC model dynamics. This representation, like all the minimal parametrizations given by three angles, has a singularity, which in the specific case occurs when $\theta = \pi/2$. In general, all these conventions should be avoided if the robot orientation is supposed to evolve in the complete $SO(3)$ manifold. However, in the particular case of the trajectories that we have tested, we safely used this representation by explicitly avoiding singular configurations for the platform pose. We chose to not use the re-arranged elements of $\mathbf{R}$ or a unit quaternion for a simple matter of convenience. Indeed, in such cases a larger state vector would have been needed. Furthermore, additional constraints, e.g., the orthogonality of the rotation matrix or the unitary-norm for the quaternion, should have been added in the resolution of the OCP, thus increasing the solver computational time and, by consequence, slowing down the available bandwidth of the controller. This can easily be dealt with, of course, by

[6] https://git.openrobots.org/projects/genom3/wiki

Preprint version,
final version at https://www.springer.com/journal/10846

4.2 Experiments with the quadrotor

using a more powerful computation unit. It should be underlined that the proposed framework does not depend on the particular orientation representation and easily adapts to the others without the need to deal with additional theoretical issues.

The cost function weights in (16) are specified at the beginning of the description of each experiment and simulation. In general, they have been chosen on a case-dependent basis taking into account heuristic considerations and often following a *trial-and-error* procedure. The automatic tuning of such weights is an important topic which is left for future work. Throughout all experiments and simulations presented in the paper, the input terms in the cost function have not been considered, i.e., the entries of the weights $\mathbf{R}_h$ related to the input are equal to zero. This has been done with the goal to exploit the MRAVs potentialities until their limits by taking advantage of the actuator dynamics up to their bounds. Therefore, we decided to test our NMPC algorithm by discarding these regularization terms. In all the performed tests, including the most agile ones, we never encountered problems in the regularity of the input evolution. Furthermore, despite the strong accelerations of some of the reference state trajectories to the NMPC algorithm, we never triggered the activation of the slack variables.

Finally, regarding the choice of the input bounds along the prediction horizon, we selected

$$\tilde{f}_{i,k+h} = f_{i,k}, \quad h = 0, \dots, N-1 \tag{34}$$

i.e., the limits are kept constant along the future window. This choice has been motivated by a matter of simplicity of implementation. A more rigorous choice could be to select the time-varying $\tilde{f}_{i,k+h}$ in relation to the predicted state evolution at the previous control step for $t = (k-1)T$. The comparison within the results produced by these two configurations is also left as future investigation.

4.2 Experiments with the quadrotor

According to the choices we made for the state and input vectors, defined by (10)-(11), and for the orientation description associated to (30), in the case of the quadrotor model we have $\mathbf{x} \in \mathbb{R}^{16}$ and $\mathbf{u} \in \mathbb{R}^4$. With this configuration, the average NMPC solver time is $t_{\text{solv}} = 3.5\text{ms}$. In the following, we present the tracking results obtained by the quadrotor with two different trajectories. The first one combines a sinusoidal chirp motion along one component of the position with a steadily horizontal and constant-heading desired orientation. Such dynamic and decoupled motion, which was designed on purpose to be dynamically unfeasible for this vehicle (see previous discussion), is also given as reference to the Tilt-Hex in order to compare the performances of the two platforms and, in particular, to highlight the different behaviors of UDT and MDT aerial robots. For a numerical comparison of the results, we refer the reader to Tab. 8. On the other hand, we designed the second reference motion in order to test the controller compliance with the actuator bounds, when dealing with a discontinuous trajectory. In particular, these tests highlight the importance of the control compliance with the input constraints for the preservation of the system stability. Also in this case, comparative numerical results are outlined in Tab. 8.

Table 6 Parameters used in the quadrotor experiments.

Parameter	Value	Unit
ν	1.2	m
ξ	0.025	$\frac{\text{rad}}{\text{s}^2}$
$\bar{t}$	44.84	s
$\boldsymbol{\epsilon}_{\mathbf{p}}$	$[-0.5\,0.2\,0.2]^\top$	m
ρ	$0.125 : 0.125 : 1.75$	[ ]
$\mathbf{Q}_{\mathbf{p}}(j,j)\|_{j=1,2,3}$	$500, 300, 300$	[ ]
$\mathbf{Q}_{\dot{\mathbf{p}}}(j,j)\|_{j=1,2,3}$	$1.9, 1.9, 1.9$	[ ]
$\mathbf{Q}_{\boldsymbol{\eta}}(j,j)\|_{j=1,2,3}$	$0.1, 0.1, 40$	[ ]
$\mathbf{Q}_{\boldsymbol{\omega}}(j,j)\|_{j=1,2,3}$	$0.15, 0.15, 0.15$	[ ]
$\mathbf{Q}_{\ddot{\mathbf{p}}}(j,j)\|_{j=1,2,3}$	$0, 0, 0$	[ ]
$\mathbf{Q}_{\dot{\boldsymbol{\omega}}}(j,j)\|_{j=1,2,3}$	$0, 0, 0$	[ ]
$\mathbf{R}_h(j,j)\|_{j=1,\dots,4}$	$0, 0, 0, 0$	[ ]

4.2.1 Position chirp trajectory

In the first experiment, the quadrotor is required to track a position reference $\mathbf{p}_r = [c(t)\,0\,0]^\top$, where the chirp signal $c(t)$ is a sine with varying frequency, with amplitude $\nu = 1.2$ m, and a triangular frequency that linearly increases from $\xi_0 = 0$ rad/s to $\xi_{\bar{t}} = 1.12$ rad/s with a slope $\xi = 0.025$ rad/s^2 in the interval $[0, \bar{t}[$, $\bar{t} = 44.84$ s, and then decreases with a slope $-\xi$ in the interval $[\bar{t}, 2\bar{t}[$. In mathematical terms, this translates into

$$c(t) = \nu \sin\big(\xi(t)\, t\big), \; \xi(t) = \begin{cases} \xi t & \text{if } t \in [0, \bar{t}[\\ \xi(\bar{t} - t) & \text{if } t \in [\bar{t}, 2\bar{t}[\end{cases} \tag{35}$$

On the other hand, the attitude reference is constantly $\boldsymbol{\eta}_r = [0\,0\,0]^\top$. Moreover, the desired position derivatives $\dot{\mathbf{p}}_r$, $\ddot{\mathbf{p}}_r$ and the rotational derivatives $\boldsymbol{\omega}_r$, $\dot{\boldsymbol{\omega}}_r$ are consistent with the definitions of $\mathbf{p}_r$ and $\boldsymbol{\eta}_r$, respectively. Regarding the form of the diagonal matrices employed to weight the different error terms inside the cost function, we used $\mathbf{Q}_k = diag(\mathbf{Q}_{\mathbf{p}}, \mathbf{Q}_{\dot{\mathbf{p}}}, \mathbf{Q}_{\boldsymbol{\eta}}, \mathbf{Q}_{\boldsymbol{\omega}}, \mathbf{Q}_{\ddot{\mathbf{p}}}, \mathbf{Q}_{\dot{\boldsymbol{\omega}}})$, $\forall k \in \{0, \dots, N\}$. The values of the trajectory parameters and the diagonal sub-blocks $\mathbf{Q}_\bullet$ chosen for the quadrotor experiments are displayed in Tab. 6. The latter ones are the result of a trial-and-error procedure that we performed, in compliance with some heuristic guidelines, in order to obtain satisfactory tracking performance. In particular, the weights associated with the orientation error have been selected much smaller than the ones related

Accepted for publication at the
Springer Journal of Intelligent & Robotic Systems, 2020

to the position error, given the impossibility for the particular platform to track the roll and pitch references. On the other hand, the yaw error has a larger impact w.r.t. the other two angular components, as the authority around this axis is still present despite the under-actuation. Finally, the feedforward terms related to $\ddot{\mathbf{p}}_d$ and $\dot{\boldsymbol{\omega}}_d$ turned out to be not very relevant in these experiments. This explains why their entries are weighted with null gains.

With reference to the desired trajectory, the platform is required (if possible) to keep a flat orientation while moving laterally along the x-axis. This motion is unfeasible for a UDT aerial vehicle, since the only way it has to steer the thrust force is by re-orienting its body chassis, with no possibility to keep it horizontal. We provided an unfeasible rotational profile on purpose with the intent of showing that the proposed NMPC scheme can manage the re-generation and tracking of a generic trajectory, subject to the limitations imposed by the particular MRAV under analysis, without the need to resort, e.g., to differential flatness. In this way, the user does not need to explicitly compute the particular platform-dependent feasible trajectory, but can delegate this task to the predictive controller, which automatically adapts the reference profile according to the robot constraints. Of course, position errors are made even smaller if a feasible reference trajectory is available and is provided to the controller. However, this was not the major point to be shown in the experiments.

The plots related to the trajectory tracking are depicted in Fig. 7. As it is visible from the first one, related to the position tracking, the trajectory is symmetric w.r.t. the time instant $t = \bar{t}$. While the position and the linear velocity are globally well tracked, the second components of the orientation and the angular velocity deviate consistently from their reference signals. This is a natural consequence of the platform inability to produce any lateral force in body frame, which causes its under-actuation. More in detail, the peaks in the measured robot pitch θ in the third plot are synchronized with the ones of the position p_x in the first plot. Indeed, the edge points on the sine corresponds to the moments of maximum lateral acceleration, which can be attained only by a re-orientation of the platform frame. With regard to the position error, illustrated in the fifth plot from the top, it is possible to observe that the negative peaks are more pronounced w.r.t. the positive ones. This asymmetry is caused by the lateral force disturbance acting on the platform due to the presence of the serial data cable, which pulls the robot in a more severe way towards the positive direction of the x-axis. The very same outcome can be consistently recognized also in the corresponding plot of Fig. 12, since the cable configuration remains unchanged throughout the experiments. Apart from the contribution of the external disturbance, the inexact position tracking is also a side effect of the unfeasible flat orientation given as reference to the predictive controller.

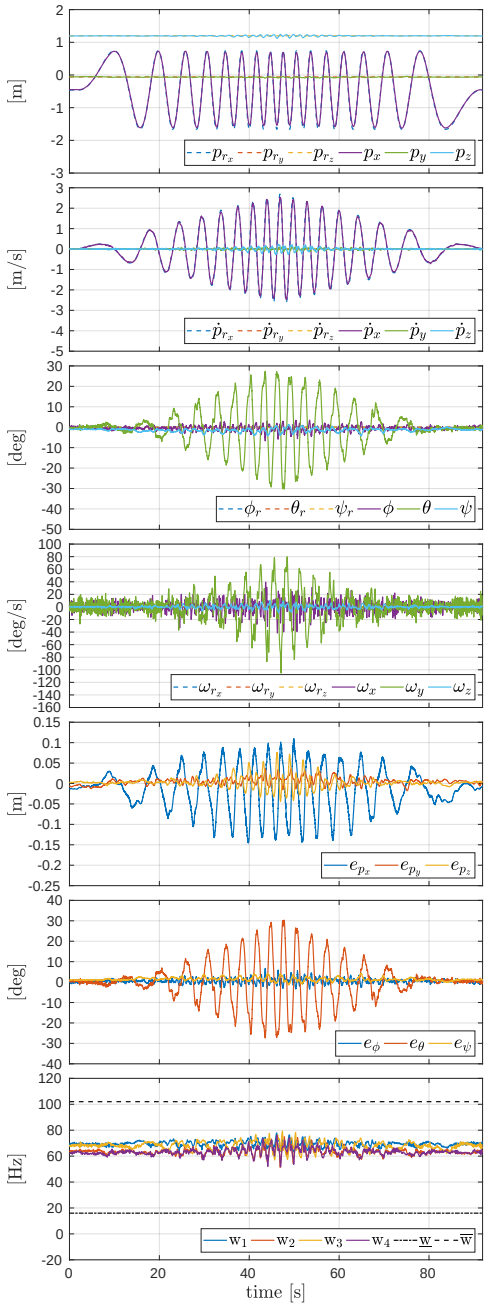

Fig. 7 Plots of the quadrotor performing a chirp trajectory on the x-axis. From top to bottom, the position, linear velocity, orientation and angular velocity tracking, the position and orientation errors, and the actuator spinning velocities.

The velocities of the MRAV rotors, whose plot is illustrated in the bottom of Fig. 7, are centered on the mean value needed to compensate the gravity force while the aerial vehicle is hovering. The small offset between the velocity of rotors 1-3 and 2-4 suggests that the serial cable also generates a small clockwise torque around the z-axis, which is

Preprint version,
final version at https://www.springer.com/journal/10846

4.2 Experiments with the quadrotor

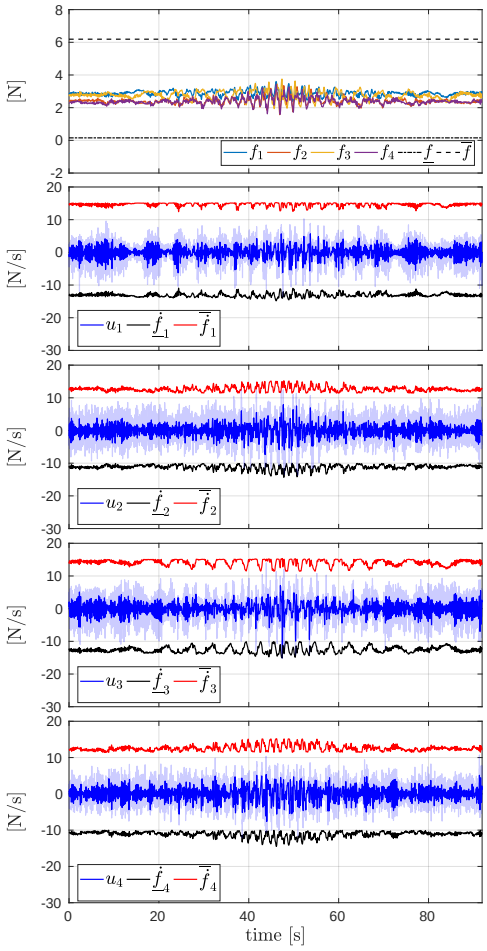

Fig. 8 Plots of the quadrotor performing a chirp trajectory on the x-axis. From top to bottom, the actuator forces and their derivatives. In particular, all the signals remain inside the feasible region delimited by the identified constraints. Notably, the noisy references u_i are overlapped by their filtered profiles.

balanced in order to keep the platform aligned with the yaw reference. In particular remark the fact that, even if the trajectory is rapidly-varying (with a linear acceleration peak of $5.85\mathrm{m/s}^2$), the rotor velocities (equivalently their produced forces, presented in the first of plot of Fig. 8) take values close to the hovering set-point, without the need to span a large set of values. This happens because the body torque needed to re-orient the aerial vehicle requires just small differences between the rotor spinning rates. As a consequence, in this experiment the actuator force derivatives do not need to assume large values. This intuition is confirmed by the plots 2-5 of Fig. 8, which show that the input components u_i, represented in blue, remain distinctively far from their lower and upper bounds, drawn in black and red, respectively. This evidence suggests that in the case of UDT-MRAVs, the limits on the input and on the state components related to the actuator forces can be reached only with rapidly-varying trajectories, designed in order to produce sudden changes in the rotor commands. This motivated the next experiment.

4.2.2 Discontinuous trajectory

Since in the chirp experiment the input limits were far from being approached, we designed a discontinuous trajectory to test the controller stability and its compliance to the actuator constraints in a critical case. For this purpose, we generated as position reference signal a sequence of steps from an initial position $\mathbf{p}_1$ to a final one $\mathbf{p}_2 = \mathbf{p}_1 + \boldsymbol{\epsilon}_\mathbf{p}$, with $\boldsymbol{\epsilon}_\mathbf{p} = [-0.5\, 0.2\, 0.2]^\top \mathrm{m}$. On the other hand, all the other reference profiles were set to zero. In this way, the vehicle was always required to reach the next hovering configuration, with an horizontal attitude and zero translational and rotational velocities, in a short time. Moreover, in order to make the experiment even more challenging, we limited on purpose the predictive capability of the controller, i.e., the NMPC algorithm was made aware about the transitions in the position reference only at the time in which such changes effectively occurred. This strategy emulated an unforeseen event against which the algorithm had to promptly and safely react. In this way, the instantaneous appearance of a consistent error in the controller easily pushed the actuator commands towards their limitations. Throughout this experiment, the identified input constraints on the actuator force derivative were re-scaled with gains ρ, taking values in $[0.125, 1.75]$, spanning from very conservative - obtained with $\rho = 0.125$ - to larger than the identified ones - obtained with $\rho = 1.75$. The input limits in the controller were manually increased, after each discontinuous motions of the robot, by the operator by means of a joystick connected to the ground station. This allowed us to empirically assess the validity of the bounds resulting from our identification.

The tracking results related to the trajectory of this specific experiment are shown in Fig. 9, where the yellow region highlights the time interval in which the enforced limits correspond exactly to the ones previously identified. The position tracking, depicted in the first plot from the top, shows that very conservative bounds for the actuators, i.e., $\rho \in [\frac{1}{8}\ \frac{1}{4}]$, cause step responses with a remarkable settling time and extended oscillations. Furthermore, the reduced capability to produce a change in the actuator forces seems to affect the tracking of the yaw, that has a non-negligible error for low values of ρ. As already ascertained in the previous experiment, this disturbance is induced by the communication cable. On the other hand, the oscillations in the step responses result much more restrained as soon as the control saturations approach the identified ones. Nevertheless, an additional increase in the control bounds imply growing overshoots, especially on the z-axis. Ultimately, the instability is reached at $t \approx 84\mathrm{s}$, when $\rho = \frac{7}{4}$. In this moment, the associated limits become almost the double of the identified

Accepted for publication at the
Springer Journal of Intelligent & Robotic Systems, 2020

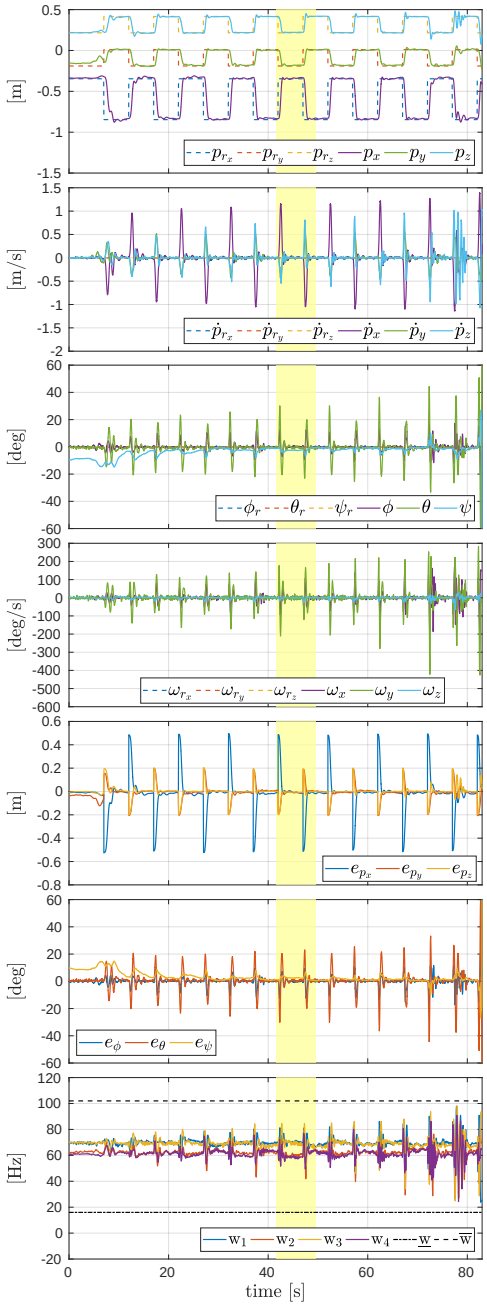

Fig. 9 Plots of the quadrotor tracking a discontinuous trajectory with steps in the position, while the controller limits are increased (the yellow region highlights the use of the identified ones). From top to bottom, the position, linear velocity, orientation and angular velocity tracking, the position and orientation errors, and the actuator spinning velocities.

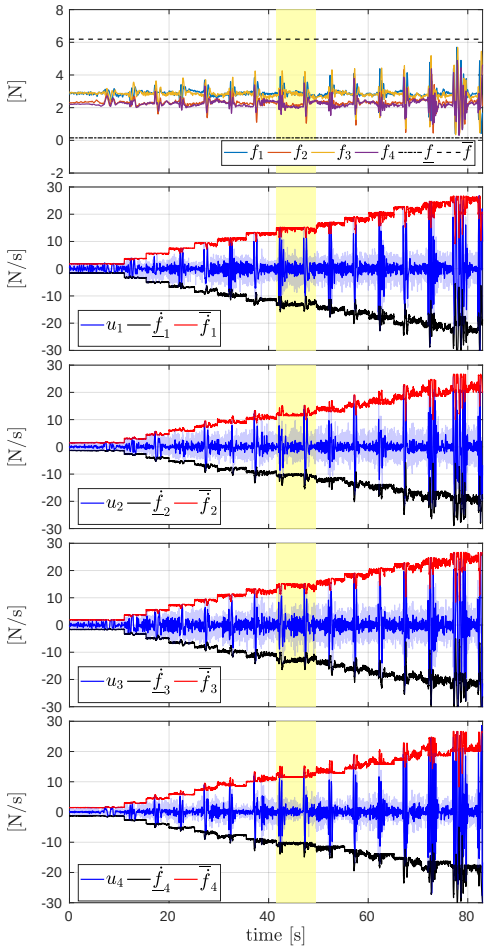

Fig. 10 Plots of the quadrotor tracking a discontinuous trajectory with steps in the position, while the controller limits are increased (the yellow region highlights the use of the identified ones). From top to bottom, the actuator forces and their derivatives. In particular, all the signals remain inside the feasible region delimited by the constraints.

ones and they induce the platform to reach a configuration from which it was not able to recover. This is confirmed by the plot of the orientation error, where the pitch error reaches almost $e_\theta = 60\,\mathrm{deg}$. The MRAV instability, which causes the experiment to abort, can be particularly well appreciated from the multimedia attachment. Regarding this point, it is worthwhile to make some considerations. First, the tracking results suggest the identified limits to be suitable to ensure the platform stability, also in such a critical experiment. Moreover, this is true within some robustness margin, which was sought in order to avoid an excessive stress for the motor currents. Finally, the plots of Fig. 10 deserve a particular attention. With reference to the first one, we can observe that the aerial vehicle becomes unstable even if the actuator forces never reach their limitations, even when instability finally happens. On the other hand, we see from the other plots that their derivatives closely approach the lower and upper bounds. This fact suggests that, neglecting the constraints on the force derivatives, as done in other works, may jeopardize, not only the system performances, but also its stability properties.

Preprint version,
final version at https://www.springer.com/journal/10846

4.3 *Experiments with the Tilt-Hex*

Table 7 Parameters used in the Tilt-Hex experiment.

Parameter	Value	Unit
ν	1.2	m
ξ	0.025	$\frac{\text{rad}}{\text{s}^2}$
$\bar{t}$	44.84	s
$\boldsymbol{\epsilon_{\mathbf{p}}}$	$[-0.4\,0.3\,0.2]^\top$	m
ρ	$0.125 : 0.125 : 1.75$	[ ]
$\mathbf{Q_p}(j,j)\|_{j=1,2,3}$	$500, 200, 200$	[ ]
$\mathbf{Q_{\dot{p}}}(j,j)\|_{j=1,2,3}$	$25, 20, 20$	[ ]
$\mathbf{Q_\eta}(j,j)\|_{j=1,2,3}$	$10, 6, 10$	[ ]
$\mathbf{Q_\omega}(j,j)\|_{j=1,2,3}$	$0.5, 0.5, 0.5$	[ ]
$\mathbf{Q_{\dot{p}}}(j,j)\|_{j=1,2,3}$	$0.01, 0.01, 0.01$	[ ]
$\mathbf{Q_{\dot{\omega}}}(j,j)\|_{j=1,2,3}$	$0, 0, 0$	[ ]
$\mathbf{R}_h(j,j)\|_{j=1,\dots,6}$	$0, 0, 0, 0, 0, 0$	[ ]

Table 8 Numerical comparison of the NMPC performance achieved in the experimental validation with different MRAVs.

Chirp trajectory		
Parameter [•]	Quadrotor	Tilt-Hex
$e_{p,\text{MAX}}$ [m]	$[0.146,\ 0.036,\ 0.076]^\top$	$[\mathbf{0.064},\ \mathbf{0.025},\ \mathbf{0.018}]^\top$
$e_{p,\text{RMS}}$ [m]	$[0.054,\ 0.010,\ 0.014]^\top$	$[\mathbf{0.015},\ \mathbf{0.006},\ \mathbf{0.007}]^\top$
$e_{\eta,\text{MAX}}$ [deg]	$[6.7,\ 30.4,\ \mathbf{4.3}]^\top$	$[\mathbf{5.1},\ \mathbf{13.3},\ 7.3]^\top$
$e_{\eta,\text{RMS}}$ [deg]	$[1.2,\ 10.0,\ 1.6]^\top$	$[\mathbf{1.2},\ \mathbf{3.9},\ \mathbf{1.4}]^\top$
$f_{i,\text{MIN}},\ f_{i,\text{MAX}}$ [N]	$1.495,\ 3.876$	$\mathbf{0.231},\ \mathbf{10.251}$
$\dot{f}_{i,\text{MIN}},\ \dot{f}_{i,\text{MAX}}$ [N/s]	$-15.264,\ 13.998$	$\mathbf{-25.320},\ \mathbf{28.321}$
Step trajectory with identified input limits ($\rho = 1$)		
Parameter [•]	Quadrotor	Tilt-Hex
$e_{\eta,\text{MAX}}$ [deg]	$[12.6,\ 30.1,\ 6.9]^\top$	$[\mathbf{4.0},\ \mathbf{5.1},\ \mathbf{2.0}]^\top$
$e_{\eta,\text{RMS}}$ [deg]	$[3.4,\ 7.9,\ 2.6]^\top$	$[\mathbf{1.1},\ \mathbf{1.1},\ \mathbf{1.2}]^\top$
$f_{i,\text{MIN}},\ f_{i,\text{MAX}}$ [N]	$1.031,\ 4.396$	$\mathbf{0.972},\ \mathbf{8.054}$
$\dot{f}_{i,\text{MIN}},\ \dot{f}_{i,\text{MAX}}$ [N/s]	$-15.162,\ 15.204$	$\mathbf{-25.348},\ \mathbf{28.103}$

4.3 Experiments with the Tilt-Hex

Compared to the quadrotor model, the one of the Tilt-Hex is characterized by two more state and input components to describe the dynamics related to the presence of the additional actuators. As a matter of fact, $\mathbf{x} \in \mathbb{R}^{18}$ and $\mathbf{u} \in \mathbb{R}^6$. With this configuration, the average NMPC solver time is $t_{\text{solv}} = 4.1\text{ms}$. In the validation campaign, we made the Tilt-Hex track both the trajectories presented in the previous experiments. The values for the cost function diagonal matrices used in this experiment are reported in Tab. 7.

4.3.1 Position chirp trajectory

Thanks to the tilting of its actuators, the Tilt-Hex can exert a 3D set of forces which is not anymore restrained to the body-frame z-axis. In particular, the polytope of forces with zero moment, computed in compliance with all the admissible actuator forces, can be appreciated from the left side of Fig. 3 in our previous work [31]. Thanks to this feature, the vehicle can track decoupled references in position and orientation. However, despite this additional capability, the Tilt-Hex cannot track any decoupled trajectory, due to the unavoidable limitations still present in the actuators.

It should be appreciated that the previously defined chirp trajectory was generated with the goal to be unfeasible also w.r.t. the Tilt-Hex actuation capabilities. In fact, by plugging

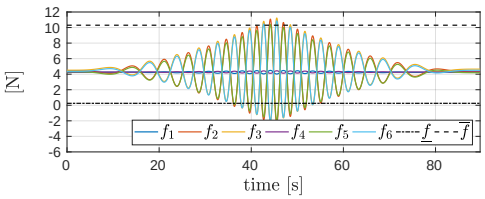

Fig. 11 Desired profile for the actuators forces obtained by inverting the model dynamics. This chirp trajectory results unfeasible also with respect to the Tilt-Hex limitations.

the desired trajectory and the physical parameters in (2) and isolating the vector $[\mathbf{f}_B^\top\ \boldsymbol{\tau}_B^\top]^\top$, we obtain the analytic expression of the wrench needed to ideally follow the 6D reference profile. At this point, inverting (6) – which is possible in this case since $\mathbf{G}$ is square and full-rank – provides the ideal (no noise or disturbance were involved in this computation) evolution of the actuator forces. As shown in Fig. 11, the desired actuator force trajectories, obtained via such dynamic inversion, are not compliant with the lower and upper bounds. This means that, also in this case, a new feasible trajectory has to be re-computed by the NMPC strategy. Nevertheless, we expect to obtain improved tracking performances compared to the quadrotor experiment.

The plots related to the trajectory tracking for this experiment are shown in Fig. 12. As shown on the two top sub-figures, the translational references are followed in a more precise way compared to Fig. 7. In particular, this is true also around the central peaks, which correspond to the most rapidly-varying part of the trajectory, i.e., where the lateral acceleration takes the largest values. From the third and the fourth plots it can be observed that the deviations from the orientation and the angular velocity references are significantly reduced w.r.t. the ones produced by the quadrotor with the very same trajectory. Such remarkable improvement is a direct consequence of the benefits induced by the multi-directionality of the thrust. On the other hand, also the position error is consistently reduced, with a maximum peak of 6.4cm (in absolute value) against the 14.5cm of the quadrotor experiment. This suggests that the full actuation also helps improving the position tracking, as already observed in our previous work. With reference to the fifth plot, the systematic small asymmetry in the position error is caused again by the cable disturbance. To ease the reader's analysis of the experimental results, we translated the graphical comparison offered by the plots into a quantitative analysis of the data, which we present in Tab. 8.

Accepted for publication at the
Springer Journal of Intelligent & Robotic Systems, 2020

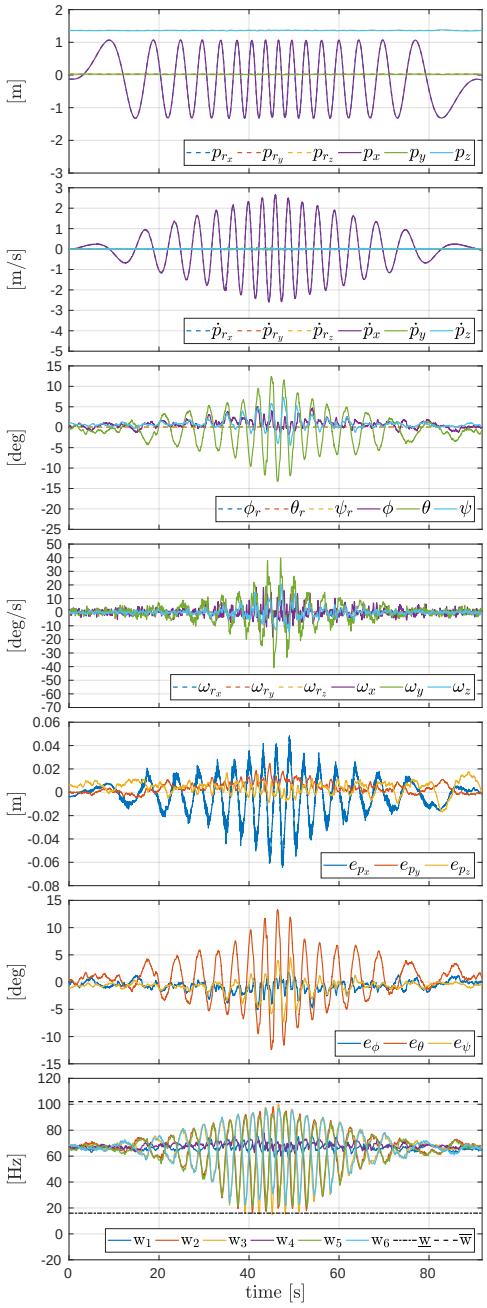

Fig. 12 Plots of the Tilt-Hex performing a chirp trajectory on the x-axis. From top to bottom, the position, linear velocity, orientation and angular velocity tracking, the position and orientation errors, and the actuator velocities.

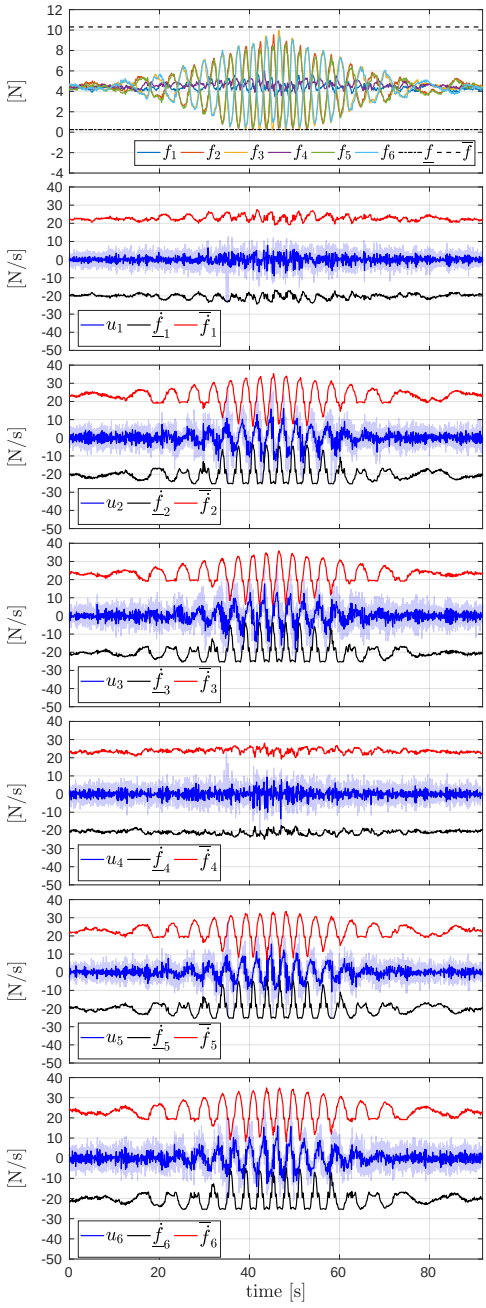

Fig. 13 Plots of the Tilt-Hex performing a chirp trajectory on the x-axis. From top to bottom, the actuator forces and their derivatives. In particular, all the signals remain inside the feasible region delimited by the constraints.

As far as the actuator data are concerned, consider the last plot of Fig. 7 and the bottom one in Fig. 12. The rotor velocities in the second case span the feasible set in a wider way. While in the quadrotor experiment the rotor speed constraints are not even approached, in the Tilt-Hex case they become frequently active. The same applies for the generated thrust forces, as shown in the first plots of Fig. 8 and Fig. 13. In the specific case, the fact that the lower bounds are reached more often than the upper ones is simply due to the platform mass. Indeed, from the first plot of Fig. 13 we can see that the mean hovering value per actuator is approximately 4N, which is closer to the lower saturation level. On the other hand, the velocities of a more massive vehicle would have approached more easily the upper part of the plot. Finally, from the other plots of Fig. 13 it should be appreciated how the Tilt-Hex force derivatives related to ac-

Preprint version,
final version at https://www.springer.com/journal/10846

4.3 *Experiments with the Tilt-Hex*

Table 9 Numerical comparison of the performance achieved in the experimental validation with the NMPC presented in this paper and the static-feedback reactive controller in [31].

Chirp trajectory		
Parameter [•]	Tiltex (CTRL [31])	Tilt-Hex (NMPC)
$e_{p,\text{MAX}}$ [m]	$[0.076,\ 0.032,\ 0.041]^\top$	$[\mathbf{0.064},\ \mathbf{0.025},\ \mathbf{0.018}]^\top$
$e_{p,\text{RMS}}$ [m]	$[0.020,\ 0.009,\ 0.013]^\top$	$[\mathbf{0.015},\ \mathbf{0.006},\ \mathbf{0.007}]^\top$
$e_{\eta,\text{MAX}}$ [deg]	$[\mathbf{2.2},\ 23.2,\ \mathbf{1.7}]^\top$	$[5.1,\ \mathbf{13.3},\ 7.3]^\top$
$e_{\eta,\text{RMS}}$ [deg]	$[\mathbf{0.8},\ 6.0,\ \mathbf{0.6}]^\top$	$[1.2,\ \mathbf{3.9},\ 1.4]^\top$
$f_{i,\text{MIN}},\ f_{i,\text{MAX}}$ [N]	$1.803,\ 6.836$	$\mathbf{0.231},\ \mathbf{10.251}$

tuators $\{2, 3, 5, 6\}$ and their state-dependent limitations oscillate in a much more dynamic way compared to the ones of the quadrotor, depicted in Fig. 8. This is due to the larger span of the feasible force set required by the controller to these actuators, as a consequence of their geometric arrangement, for the tracking of the same trajectory.

We now shortly compare the results achieved by this NMPC algorithm with the ones obtained by the reactive static-feedback controller designed in our previous work [31]. In particular, Fig. 5 of [31] presents the tracking results related to the same trajectory and the same MRAV. To provide a better mean for the reader to appreciate the experimental results, we report a quantitative comparison of the data obtained with the two controllers in Tab. 9. Regarding the position error, we achieved a reduced root mean square position (RMS) error with the NMPC regulator, in particular in the two lateral tails of the trajectory, where the error is always bounded within 4cm). Furthermore, while the error profile obtained with the reactive controller was more or less uniformly distributed along the trajectory, in the present case its trend seems to be proportional to the chirp frequency, which also has a triangular envelope. This effect could be explained by the predictive nature of the algorithm discussed in this paper. Indeed, while the reactive regulator always acts in relation to the instantaneous value of the desired trajectory, the NMPC response is affected by the future evolution of the former, which depends on the chirp frequency. As far as the orientation tracking is concerned, a relevant improvement is achieved. As a matter of fact, the maximum pitch error is reduced from $23\,\text{deg}$ to $13\,\text{deg}$, i.e., a decrease of more than 43%. Furthermore, analyzing the plot of the rotor velocities we see that now they evolve in a larger range, meaning that the NMPC regulator is exploiting the actuator capabilities in a more efficient way. This is a consequence of the fact that the previous controller deals with a less precise – and more conservative – model of the platform.

4.3.2 *Discontinuous trajectory*

To assess the effectiveness of our procedure in identifying meaningful actuator limitations for a non-specific hardware setup, we replicated the experiment described in 4.2.2 using

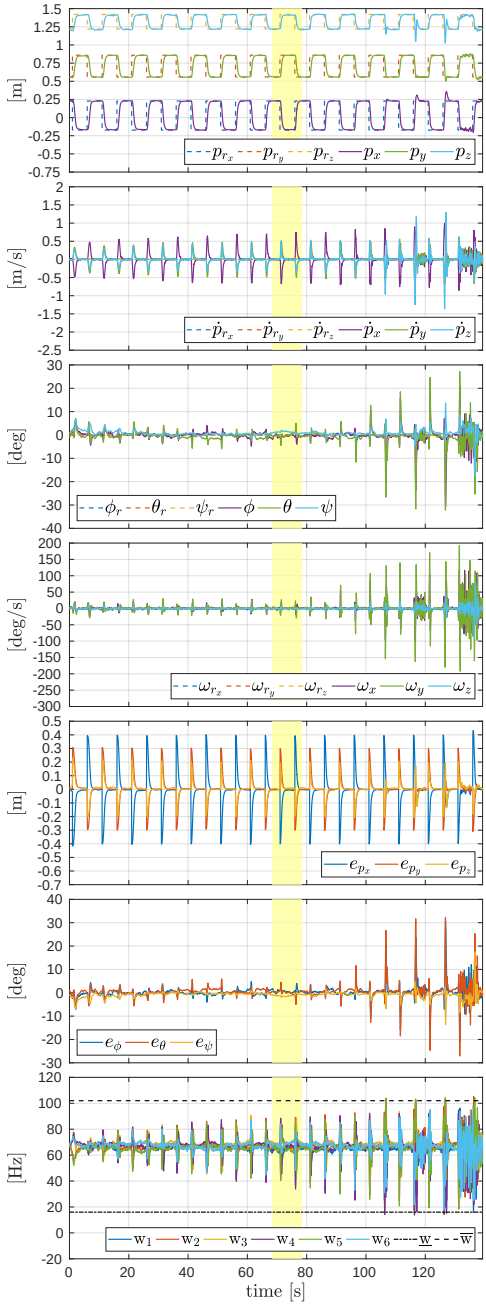

Fig. 14 Plots of the Tilt-Hex tracking a discontinuous trajectory with steps in the position, while the controller limits are increased (the yellow region highlights the use of the identified ones). From top to bottom, the position, linear velocity, orientation and angular velocity tracking, the position and orientation errors, and the actuator spinning velocities.

the Tilt-Hex robot. The plots related to this test are depicted in Fig. 14 and in Fig. 15. For this experiment, the limits to the NMPC were scaled by the user after two consecutive jumps of the MRAV, while $\boldsymbol{\epsilon_p} = [-0.5\,0.3\,0.2]^\top\,\text{m}$. The experiment outcomes show that the best step responses are achieved when the actuator limits are closer to the identified ones. This confirms again the validity of our approach. Fur-

Accepted for publication at the
Springer Journal of Intelligent & Robotic Systems, 2020

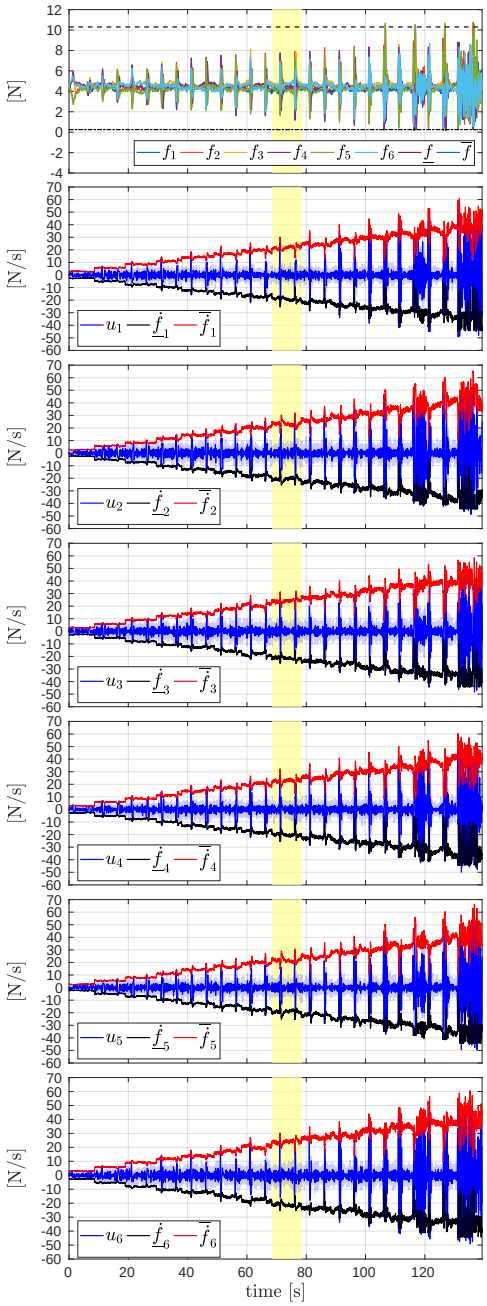

Fig. 15 Plots of the Tilt-Hex tracking a discontinuous trajectory with steps in the position, while the controller limits are increased (the yellow region highlights the use of the identified ones). From top to bottom, the actuator forces and their derivatives. In particular, all the signals remain inside the feasible region delimited by the constraints.

thermore, also in this case, the instability is reached when $\rho = \frac{7}{4}$, i.e., when the force derivative bounds are almost the double of the identified ones, which shows an adequate margin of conservativeness for the chosen limits. Also for this experiment, a numerical comparison of the most interesting results obtained with the two aforementioned aerial vehicles driven with the identified limits ($\rho = 1$), is provided in Tab. 8. Given the discontinuous nature of the reference position trajectory, it is not very meaningful to analyze the maximum or the RMS position error, in this case. On the other hand, it is much more interesting to compare the orientation tracking in the two cases. While the UDT platform has to consistently deviate from the reference flat-hovering orientation in order to generate the needed lateral force to track the position step, the MDT robot almost does not need any re-orientation of its chassis. Such remarkable effect can be visually appreciated in the attached multimedia content, where the motion of the two MRAVs have been juxtaposed. To conclude, we highlight that also in this case a bigger span of the feasible rotor velocities is obtained with the MDT MRAV.

Fig. 16 Photos of the FAST-Hex (left) and the rotor failed Tilt-Hex (right).

5 Validation with real-time simulations

In order to support the claim that our framework can deal with a generic MRAV design, we provide additional numerical validations with two other different vehicle models, shown in Fig. 16. The first one, depicted on the left, is called FAST-Hex, i.e., *Fully-Actuated by Synchronized Tilting propellers hexarotor*. This original MRAV, introduced in our previous work [60], has the capability of synchronously modifying the orientation of its actuators, thanks to a single additional servo-motor. Exploiting this further degree of freedom, the FAST-Hex can actively regulate the angle α, c.f. Fig. 1, allowing the robot to pass from an UDT configuration to an MDT one, and conversely. Moreover, the value of α can be automatically regulated, i.e., without the need of an external planner. The physical parameters of the FAST-Hex are condensed in Tab. 10.

The second vehicle, shown on the right of Fig. 16, is a pentarotor (a multi-rotor with five propellers) obtained as a failed Tilt-Hex MRAV, i.e., the platform already described in the experimental validation, but after a rotor failure. In particular the $6-th$ rotor is not allowed to spin, due to, e.g., a technical problem, and cannot exert a thrust force and generate a drag torque. For this reason, from a control point of view, we consider that such actuator is not present. The rotor failure essentially modifies the available set of body forces and torques. As already pointed out in previous contributions, in case $\alpha = \beta = 0$ it is not possible with five uni-directional actuators to generate torques in pitch and roll without generating a residual disturbing torque in the

Preprint version,
final version at https://www.springer.com/journal/10846

5.1 Simulations with the FAST-Hex

Table 10 Physical parameters of the FAST-Hex.

FAST-Hex		
Parameter	Value	Unit
m	2.4	Kg
$\mathbf{J}(:,1)$	$[0.042\ 0\ 0]^\top$	Kg m^2
$\mathbf{J}(:,2)$	$[0\ 0.042\ 0]^\top$	Kg m^2
$\mathbf{J}(:,3)$	$[0\ 0\ 0.083]^\top$	Kg m^2
c_i	$(-1)^i$	[ ]
c_f^τ	1.9e−2	m
c_f	9.9e−4	N/Hz^2
$\mathbf{R}_{A_i}^B$	$\mathbf{R}_z\left((i-1)\frac{\pi}{3}\right)\mathbf{R}_x(\alpha_i)\mathbf{R}_y(\beta)$	[ ]
$\mathbf{p}_{A_i}^B$	$\mathbf{R}_z\left((i-1)\frac{\pi}{3}\right)[\ell\ 0\ 0]^\top$	[ ]
α_i	$(-1)^{i-1}\lvert\alpha\rvert$	deg
β	0	deg
ℓ	0.315	m

yaw axis, cf. [61, 62]. In the more general case in which $(\alpha, \beta) \neq (0, 0)$, however, the platform maintains the ability to hover, see [62]. Nevertheless, the hovering orientation can not be flat any more, and depends of the actuator tilting angles, cf. [51]. We show that the NMPC controller can satisfactorily deal with the problem of static hovering, without the need to a-priori compute the steady-state orientation.

5.1 Simulations with the FAST-Hex

In order to take into account the evolution of the angle α and to let the NMPC algorithm manage its automatic regulation, we expanded the state and the input vector, defined in (10) and in (11), in the following way

$$\mathbf{x} := \left[\mathbf{p}^\top\ \dot{\mathbf{p}}^\top\ \boldsymbol{\eta}^\top\ \boldsymbol{\omega}^\top\ \boldsymbol{\gamma}^\top, \alpha\right]^\top \tag{36}$$

$$\mathbf{u} := \left[\dot{\boldsymbol{\gamma}}, \dot{\alpha}\right] \tag{37}$$

The angle α is now a component of the state vector, while $\dot{\alpha}$ is regarded as an additional control input to be optimized at each control iteration. This allows to constrain the synchronous tilting angle and its derivative within their feasible sets, computed accordingly to the data of the real MRAV prototype designed in [60]. According to this choice, for the FAST-Hex model we have $\mathbf{x} \in \mathbb{R}^{19}$ and $\mathbf{u} \in \mathbb{R}^7$.

As it can be appreciated from Fig. 3 in [60], the larger the angle α, the larger the set of body-frame lateral forces. This translates also into the possibility of decoupling the control of the body force and moment in a larger extent, which becomes particularly useful in many realistic scenarios, ranging from 6D trajectory tracking, see [31], to aerial physical interaction tasks, see [2,3], and disturbance rejection in general. On the other hand, the increase of the tilting angle implies also an increment in the energy consumption. In fact, the progressive decrease in the projection of the thrust vector along $\mathbf{z}_W$ must be compensated by an increase in the thrust intensity. In view of these considerations, it might be beneficial to regulate the angle α w.r.t. the particular task to be accomplished, while trying to minimize the energy consumption. In order to fulfill this requirement, we expanded also the output vector as follows

$$\mathbf{y}(t) = \mathbf{h}\left(\mathbf{x}(t), \mathbf{u}(t)\right) = \begin{bmatrix} \mathbf{p}(t) \\ \dot{\mathbf{p}}(t) \\ \ddot{\mathbf{p}}\left(\mathbf{x}(t), \mathbf{u}(t)\right) \\ \boldsymbol{\eta}(t) \\ \boldsymbol{\omega}(t) \\ \dot{\boldsymbol{\omega}}\left(\mathbf{x}(t), \mathbf{u}(t)\right) \\ c_e\left(\mathbf{x}(t), \mathbf{u}(t)\right) \end{bmatrix} \tag{38}$$

where the cost related to power consumption is taken into account using the following additional cost

$$c_e\left(\mathbf{x}(t), \mathbf{u}(t)\right) = \sum_{i=1}^{n} f_i^2 \tag{39}$$

which is integrated along the prediction horizon. Such model has been chosen mainly due to its simple dependency on the state components f_i, but other models can be employed.

In this context, we target the classical problem of trajectory regulation to a certain 6D configuration, i.e., the flat hovering, adding the effect of an external unknown disturbance from the environment, which emulates, in a simplified but meaningful way, the scenario of a physical interaction task or an external wind. In the first simulation, we exploit the possibility of regulating α. In this way we show how our NMPC algorithm can automatically and actively manipulate the additional control input $\dot{\alpha}$, thus improving our previous work [60]. Furthermore, in order to demonstrate the usefulness of this supplementary degree of freedom, we present the results of the same simulation, where the tilting angle α is forced to assume different fixed values and cannot be regulated.

In order to make the simulations more realistic, we added to the measured state a noise, obtained by filtering a zero-mean white Gaussian noise with a first-order causal low-pass filter having a cut-off frequency Ω_{filt}, whose value has been estimated analyzing real experimental data. The noise standard deviation values $\boldsymbol{\sigma}_\bullet$ are collected in Tab. 11, together with the other trajectory parameters, state/input bounds and cost function weights. In particular, the values of $\boldsymbol{\sigma}_\bullet$ are related to very unfavorable conditions compared to the use of typical sensors such as MoCap and gyros.

5.1.1 Hovering trajectory with unknown lateral force disturbance

Alongside the presented simulations, the FAST-Hex is required to hover maintaining a flat orientation, i.e.,

$$\mathbf{p}_r = [0.6\ 0.6\ 0.75]^\top,\ \dot{\mathbf{p}}_r = \ddot{\mathbf{p}}_r = [0\ 0\ 0]^\top$$
$$\boldsymbol{\eta}_r = \boldsymbol{\omega}_r = \dot{\boldsymbol{\omega}}_r = [0\ 0\ 0]^\top \tag{40}$$

Accepted for publication at the
Springer Journal of Intelligent & Robotic Systems, 2020

Table 11 Parameters used in the FAST-Hex simulation.

Parameter	Value	Unit
$\boldsymbol{\sigma}_{\mathbf{p}}$	$[\sqrt{0.005}\ \sqrt{0.005}\ \sqrt{0.005}]^\top$	m
$\boldsymbol{\sigma}_{\dot{\mathbf{p}}}$	$[\sqrt{0.02}\ \sqrt{0.02}\ \sqrt{0.02}]^\top$	m/s
$\boldsymbol{\sigma}_{\boldsymbol{\eta}}$	$[\sqrt{1}\ \sqrt{1}\ \sqrt{1}]^\top$	deg
$\boldsymbol{\sigma}_{\boldsymbol{\omega}}$	$[\sqrt{0.15}\ \sqrt{0.15}\ \sqrt{0.05}]^\top$	deg/s
Ω_{filt}	25	rad/s
t_1	10	s
t_2	20	s
$\mathbf{f}_{\text{dist}}(t_2)$	$3\ [cos(\frac{\pi}{3})\ sin(\frac{\pi}{3})\ 0]^\top$	N
$\underline{\alpha}$, $\overline{\alpha}$	-35 , 35	deg
$\underline{\dot{\alpha}}$, $\overline{\dot{\alpha}}$	-8.75 , 8.75	deg/s
$\mathbf{Q}_{\mathbf{p}}(j,j)\|_{j=1,2,3}$	$50, 50, 50$	[ ]
$\mathbf{Q}_{\dot{\mathbf{p}}}(j,j)\|_{j=1,2,3}$	$0.5, 0.5, 0.5$	[ ]
$\mathbf{Q}_{\boldsymbol{\eta}}(j,j)\|_{j=1,2,3}$	$15, 15, 15$	[ ]
$\mathbf{Q}_{\boldsymbol{\omega}}(j,j)\|_{j=1,2,3}$	$0.01, 0.01, 0.01$	[ ]
$\mathbf{Q}_{\ddot{\mathbf{p}}}(j,j)\|_{j=1,2,3}$	$0.0001, 0.0001, 0.0001$	[ ]
$\mathbf{Q}_{\dot{\boldsymbol{\omega}}}(j,j)\|_{j=1,2,3}$	$0, 0, 0$	[ ]
$\mathbf{R}_h(j,j)\|_{j=1,\dots,7}$	$0, 0, 0, 0, 0, 0, 0$	[ ]
Q_{e_c}	0.0005	[ ]

under the effect of a lateral force disturbance $\mathbf{f}_{\text{dist}}$ with a triangular profile. Such force, unknown to the controller, has a triangular shape from t_1 to $t_1 + t_2$, with a peak in module of 3 N at t_2, while it is $\mathbf{f}_{\text{dist}} = \mathbf{0}$ N elsewhere. As the steady-state orientation is not known a priori, the reference of the energetic term $c_{e,r}$ is constantly equal to the one needed for hovering horizontally with $\alpha = 0$, i.e., $c_{e,r} = \sum_{i=1}^{n}(\frac{mg}{6})^2$.

As long as the disturbance is not active, the NMPC algorithm should try to maintain α small, ideally equal to zero. This claim is motivated by the fact that this trajectory does not need the MDT capability in order to be tracked. On the other hand, as soon as the lateral force is activated, the platform can react to it either tilting its actuators or re-orienting its chassis. In this choice, the relative values of the cost function weights play a fundamental role. Intuitively, if the energy cost is weighted consistently (w.r.t. the tracking error terms on the states), the control algorithm should try to produce an input with low energy consumption, giving less priority to the trajectory tracking. In particular, the task of maintaining a flat orientation should be somehow discouraged by the controller, since the generation of a lateral force in this configuration would require a consistent increase of some of the actuator forces, thus raising up the energy consumption. Conversely, if the weight related to the energy cost is small, the controller would always privilege the trajectory tracking, acting on input α.

In the first simulation, related to the case in which α is actively regulated, we try to achieve a good *trade-off* between the two tasks. In the other simulations, corresponding to different fixed configurations for α, all the parameters are left untouched, in order to fairly compare the resulting performance, in terms of the overall cost function, w.r.t. the variable case.

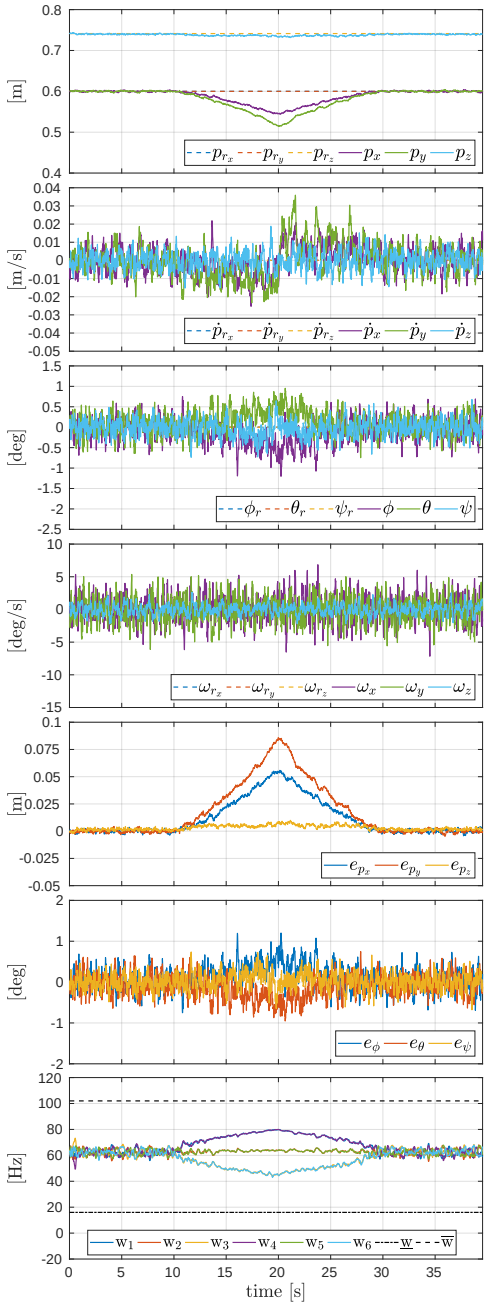

Fig. 17 Plots of the FAST-Hex (with variable α regulated from the MPC algorithm) while hovering. The robot is disturbed with an external lateral force with a triangular profile. From top to bottom, the position, linear velocity, orientation and angular velocity tracking, the position and orientation errors, and the actuator spinning velocities.

The plots related to the trajectory tracking in the variable case are depicted in Fig. 17. The first four plots, exhibiting the trajectory tracking of the state components, outline the good performance of the controller. Indeed, the measured linear velocity, orientation, and angular velocity tracking keep very close to their reference profile, which are constantly equal to zero on all components. On the other hand, the measured position visibly deviates from the reference

Preprint version,
final version at https://www.springer.com/journal/10846

5.1 Simulations with the FAST-Hex

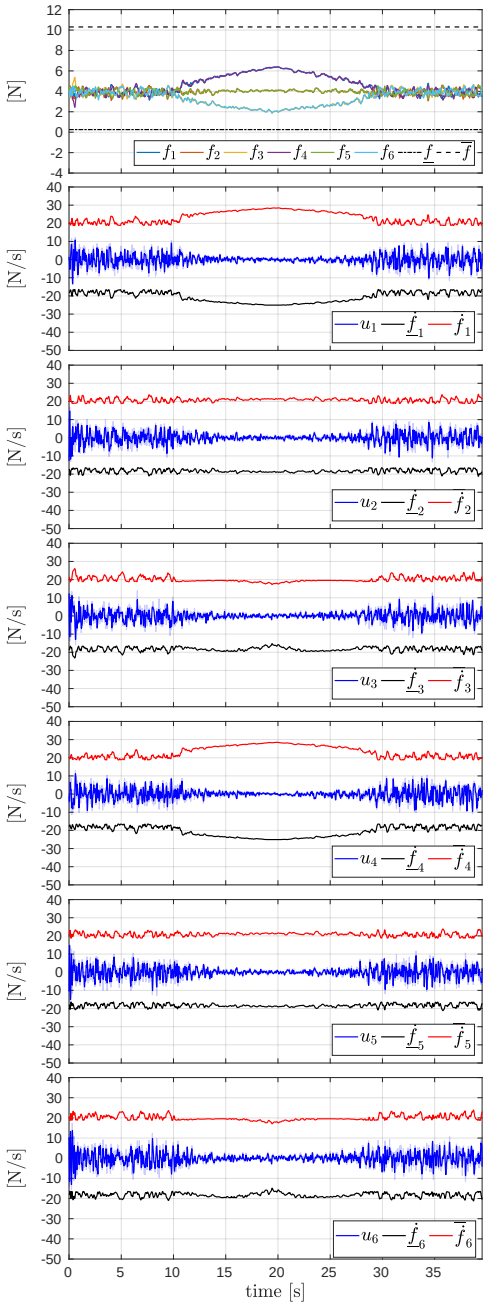

Fig. 18 Plots of the FAST-Hex (with variable α regulated from the MPC algorithm) while hovering. The robot is disturbed with an external lateral force with a triangular profile. From top to bottom, the actuator forces and their derivatives. In particular, all the signals remain inside the feasible region delimited by the constraints.

one when the disturbance is acting on the robot. Nevertheless, the position error keeps bounded, with a peak of less than $9\,\mathrm{cm}$ on its second component, which corresponds to the direction mostly affected by the lateral force, as shown in the last plot of Fig 19. This error could be considerably reduced by increasing the relative weights inside the NMPC algorithm cost function: this is confirmed by the sixth plot of Fig. 17, where the MRAV maintains the orientation error

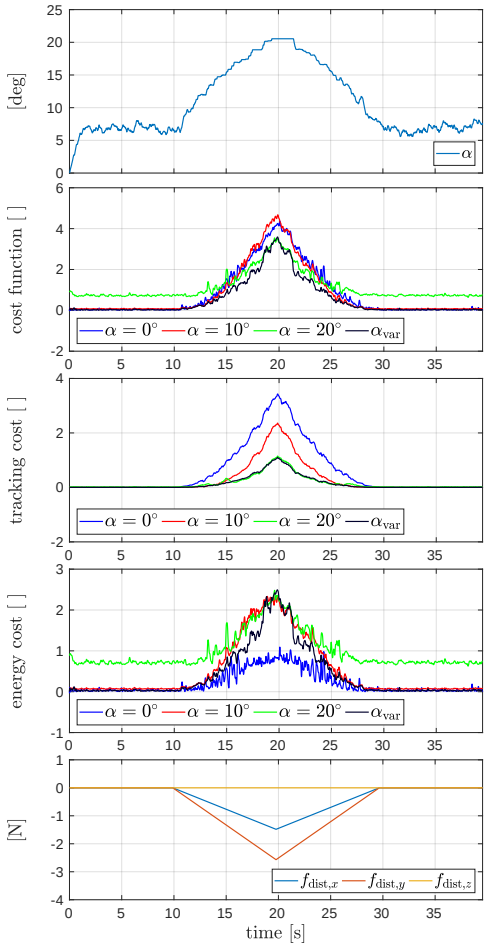

Fig. 19 Plots of the FAST-Hex while hovering. In the first two plots, the evolution of α in the variable case and the comparison of the total cost function for different cases of constant α and the variable α (regulated from the MPC algorithm). Then, the comparison of the partial costs related to the tracking and the energy terms. Finally, the profile of the external force disturbance (in World Frame).

below $1\,\mathrm{deg}$. We were able to achieve such result by properly weighting the attitude term in relation to the others, in particular in relation to the energy cost. Moreover, the last plot of the same figure shows that the external disturbance can be counteracted without an excessive effort of the rotors, since their spinning velocities (and so the generated forces) safely remain with the bounds. The plots related to the force derivatives, which are presented in Fig. 18, confirm that a static trajectory, combined with a slowly-varying disturbance, does not produce large values for the inputs.

Consider the first plot of Fig. 19, which depicts the trajectory of α. During the middle phase, the tilting angle is increased up to $\approx 21\,\mathrm{deg}$ in order to counteract the lateral force and to keep the platform flat at the same time. On the other hand, the reason why α is regulated to a constant value of $\approx 7\,\mathrm{deg}$ and not exactly to zero, is due to the noise introduced in the simulation, in particular to the one related to the translational part of the state $[p_x\ p_y\ \dot{p}_x\ \dot{p}_y]^\top$. Indeed,

Accepted for publication at the
Springer Journal of Intelligent & Robotic Systems, 2020

the control algorithm is informed about a non-zero error in these components, and continuously tries to annihilate it by selecting a small tilting angle, in order to be able to exert a lateral force and stay horizontal at the same time.

In order to demonstrate the benefit of the active regulation of the tilting angle, we additionally performed other three simulations (with the same parameters) imposing $\alpha = 0, 10, 20 \ \text{deg}$, respectively. The comparison of the overall NMPC cost functions for the different fixed cases and the variable one is displayed in the second plot of Fig. 19. As it can be appreciated, the regulated case, denoted with α_{var}, gives the best trade-off between tracking performance and consumed energy. In the unperturbed hovering phases (lateral parts of the plots), α is regulated to a small value in order to avoid unnecessary energy waste, while in the middle phase, when the disturbance force is activated, α is increased in order to improve the trajectory tracking, in particular the one related to the orientation. The third and the fourth plots of the same figure outline the partial costs related to the tracking errors and the energy cost. Among the fixed configurations, the one with the largest tilting angle, i.e. $\alpha = 20\,\text{deg}$, generates the smallest tracking cost along all the simulation. This confirms that the MDT capability drastically improves the MRAV tracking performance. However, it unavoidably causes a larger energy cost as the angle takes larger values. This is why the additional degree of freedom on α might be very convenient in many applications.

5.2 Simulations of the Tilt-Hex with rotor failure

The problem of the robustness of a MRAV in case of a rotor failure is not new in the literature. Indeed, the analysis and the design of a tilted-rotor hexarotor for fault tolerance has been considered in [62], while formal definitions as well as the design of an analytic controller based on the identification of a direction in the force space, along which the intensity of the control force can be assigned independently from the torque, can be found in [50, 51]. Given the importance of such topic in the aerial robotics panorama, we decided to target this problem, showing that our NMPC algorithm can deal with this problem in a very efficient and general way.

The failure of one rotor in the Tilt-Hex is modeled removing one state and one input, i.e., those related to the $6-th$ actuator. As a matter of fact, $\mathbf{x} \in \mathbb{R}^{17}$ and $\mathbf{u} \in \mathbb{R}^5$ in this case. In the following, we present the hovering performance in two different configurations of the angle β, c.f. Fig. 1, in order to highlight the importance of such angle in relation to the fault tolerance capabilities, c.f. [51]. The parameters related to these simulations are reported in Tab. 12.

As already pointed out in [50], given the particular arrangement of the Tilt-Hex actuators, which are symmetrically disposed in a star-configuration with alternated α and equal β angles, it is convenient to switch off the actuator located in the mirrored position w.r.t. the broken one, when the failure is detected. In this case, this corresponds to the $3-rd$ one. This choice represents the best solution in order to balance the control effort, c.f. [50], Fig. 3. In the following simulations, this behavior is emulated by setting $\underline{f} = 0\text{N}$, i.e., letting the controller the possibility to completely switch off the actuators.

Table 12 Parameters used in the rotor-failed Tilt-Hex simulation.

Parameter	Value	Unit
$\boldsymbol{\sigma}_{\mathbf{p}}$	$[\sqrt{0.005}\ \sqrt{0.005}\ \sqrt{0.005}]^\top$	m
$\boldsymbol{\sigma}_{\dot{\mathbf{p}}}$	$[\sqrt{0.02}\ \sqrt{0.02}\ \sqrt{0.02}]^\top$	m/s
$\boldsymbol{\sigma}_{\boldsymbol{\eta}}$	$[\sqrt{1}\ \sqrt{1}\ \sqrt{1}]^\top$	deg
$\boldsymbol{\sigma}_{\boldsymbol{\omega}}$	$[\sqrt{0.15}\ \sqrt{0.15}\ \sqrt{0.05}]^\top$	deg/s
Ω_{filt}	25	rad/s
t_1	5	s
$\boldsymbol{\tau}_{\text{dist}}$	$\frac{1}{250}\,[0.68\ 0.39\ 0.62]^\top$	Nm
$\mathbf{Q}_{\mathbf{p}}(j,j)\|_{j=1,2,3}$	$10, 10, 10$	[ ]
$\mathbf{Q}_{\dot{\mathbf{p}}}(j,j)\|_{j=1,2,3}$	$0.5, 0.5, 0.5$	[ ]
$\mathbf{Q}_{\boldsymbol{\eta}}(j,j)\|_{j=1,2,3}$	$1.5, 1.5, 1.5$	[ ]
$\mathbf{Q}_{\boldsymbol{\omega}}(j,j)\|_{j=1,2,3}$	$0.0005, 0.0005, 0.0005$	[ ]
$\mathbf{Q}_{\dot{\mathbf{p}}}(j,j)\|_{j=1,2,3}$	$0, 0, 0$	[ ]
$\mathbf{Q}_{\boldsymbol{\omega}}(j,j)\|_{j=1,2,3}$	$0, 0, 0$	[ ]
$\mathbf{R}_h(j,j)\|_{j=1,\dots,5}$	$0, 0, 0, 0, 0$	[ ]

5.2.1 Hovering trajectory with unknown torque disturbance

In this case, the reference trajectory is again a static hovering,

$$\mathbf{p}_r = [0\ 0\ 0.75]^\top,\ \dot{\mathbf{p}}_r = \ddot{\mathbf{p}}_r = [0\ 0\ 0]^\top$$
$$\boldsymbol{\eta}_r = \boldsymbol{\omega}_r = \dot{\boldsymbol{\omega}}_r = [0\ 0\ 0]^\top$$

In order to make the simulations even more realistic, in addition to the already introduced measurement noise, we add a torque disturbance to the platform, whose magnitude can be compared to typical values that one could experience in a real experiment due to parameter mismatches and/or external perturbations. The way this torque $\boldsymbol{\tau}_{\text{dist}}$ is computed deserves some explanations. In the case $\beta = 0$, when both the $6-th$ and the $3-rd$ actuators are switched off, the moments generated by the other four propellers lie all on a 2-dimensional plane, c.f. [51], Fig. 3. This can be verified by analyzing the rank of the allocation sub-matrix ${}^3\mathbf{G}_2^6 = \mathbf{G}_2(:, 1, 2, 4, 5)$, i.e., the sub-part related to the torque actuation deprived of the columns related to the actuators which are broken (the $6-th$) and off (the $3-rd$), respectively. At this point, we select the normal to such plane by finding an orthonormal base $\{\mathbf{v}_1\ \mathbf{v}_2\}$ for the column span of ${}^3\mathbf{G}_2^6$ and operate the cross product $\mathbf{v}_3 = \mathbf{v}_1 \times \mathbf{v}_2$. This unit vector indicates the direction of the most unfavorable torque disturbance for the platform when $\beta = 0$ and only actuators $\{1, 2, 4, 5\}$ are effectively working. In order ensure that such perturbation cannot be compensated by a MRAV

Preprint version,
final version at https://www.springer.com/journal/10846

5.2 Simulations of the Tilt-Hex with rotor failure

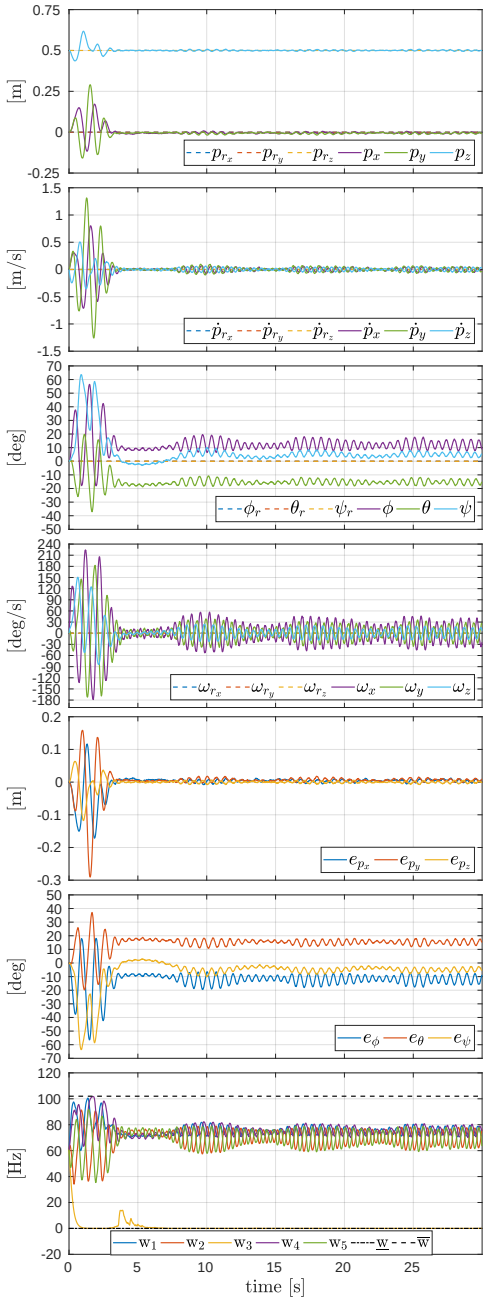

Fig. 20 Plots of the Tilt-Hex with rotor failure and $\beta = 0\,\text{deg}$ while hovering. The robot is disturbed with a constant external torque. From top to bottom, the position, linear velocity, orientation and angular velocity tracking, the position and orientation errors, and the actuator spinning velocities.

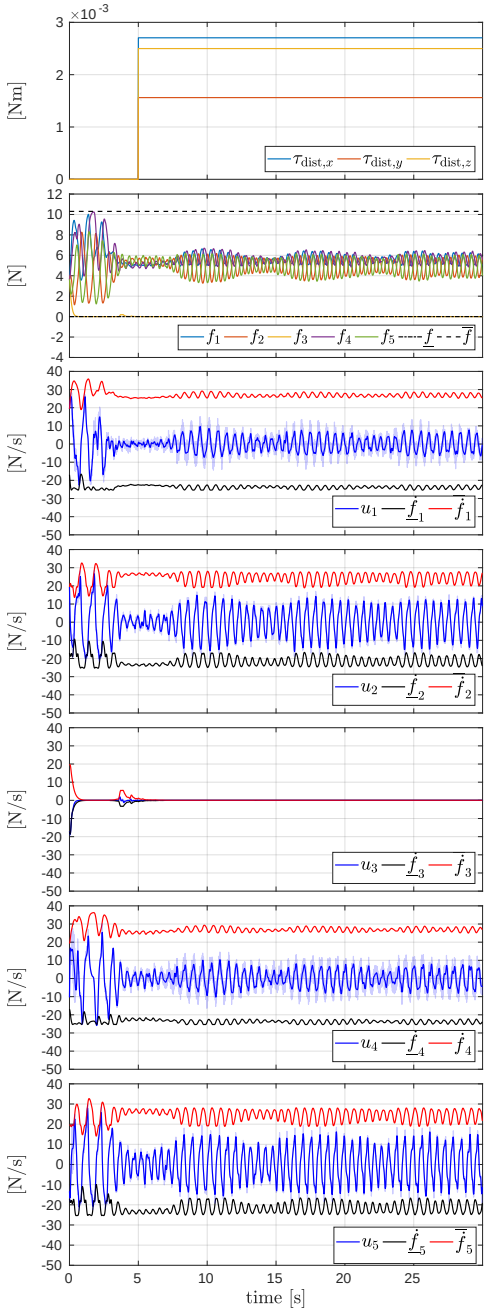

Fig. 21 Plots of the Tilt-Hex with rotor failure and $\beta = 0\,\text{deg}$ while hovering. The robot is disturbed with a constant external torque $\boldsymbol{\tau}_{\text{dist}}$, activated at $t = t_1$. From top to bottom, the disturbance torque, the actuator forces and their derivatives. In particular, all the signals remain inside the feasible region delimited by the constraints.

with this tilting configuration, even if the $3 - rd$ actuator is actively used, we verify that $\mathbf{v}_3$ has a positive projection along the direction of the total torque $\boldsymbol{\tau}_3^B$ that can be generated by such actuator. In mathematical terms, we select $\mathbf{v}_3' = \text{sgn}(\mathbf{v}_3^\top \boldsymbol{\tau}_3^B)\mathbf{v}_3$. Finally, we scale down the vector norm in order to obtain a meaningful order of magnitude for the disturbance, i.e., $\boldsymbol{\tau}_{\text{dist}} = \frac{1}{250}\mathbf{v}_3'$. In the presented simulations, it is activated at $t = t_1$. The evolution of such perturbation, constant in body frame, is depicted in the first plots of Fig. 21 and Fig. 23.

The plots of this simulation related to the case $\beta = 0\,\text{deg}$ are depicted in Fig. 20 and in Fig. 21, while the ones obtained with $\beta = -25\,\text{deg}$ are portrayed in Fig. 22 and in Fig. 23. Comparing both the position and the orientation errors in the two cases, we can see that for $\beta = 0\,\text{deg}$ the platform cannot hover statically, since it periodically oscil-

Accepted for publication at the
Springer Journal of Intelligent & Robotic Systems, 2020

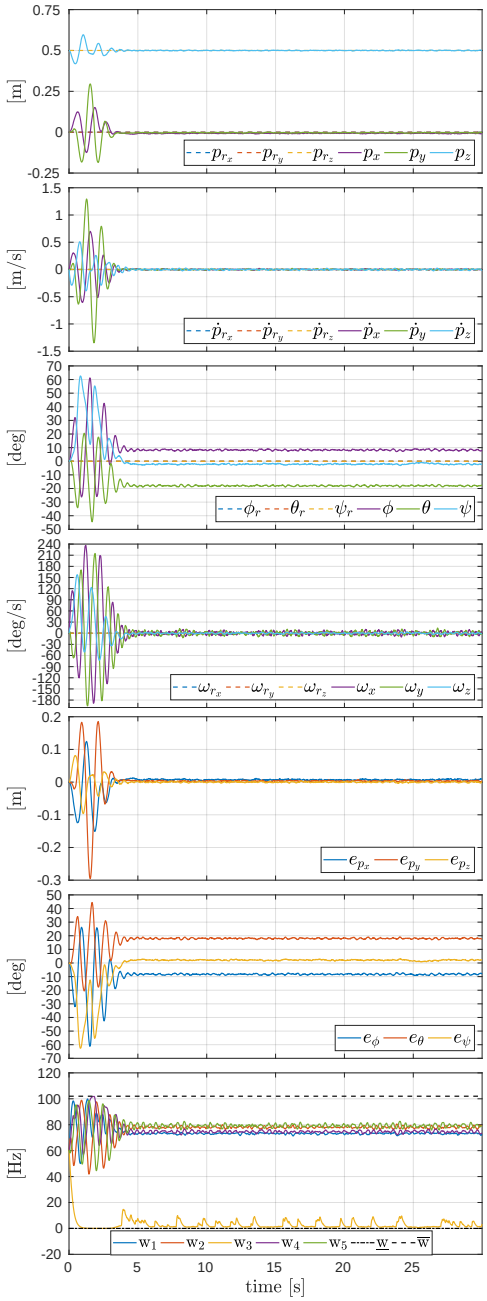

Fig. 22 Plots of the Tilt-Hex with rotor failure and $\beta = -25\,\text{deg}$ while hovering. The robot is disturbed with a constant external torque. From top to bottom, the position, linear velocity, orientation and angular velocity tracking, the position and orientation errors, and the actuator spinning velocities.

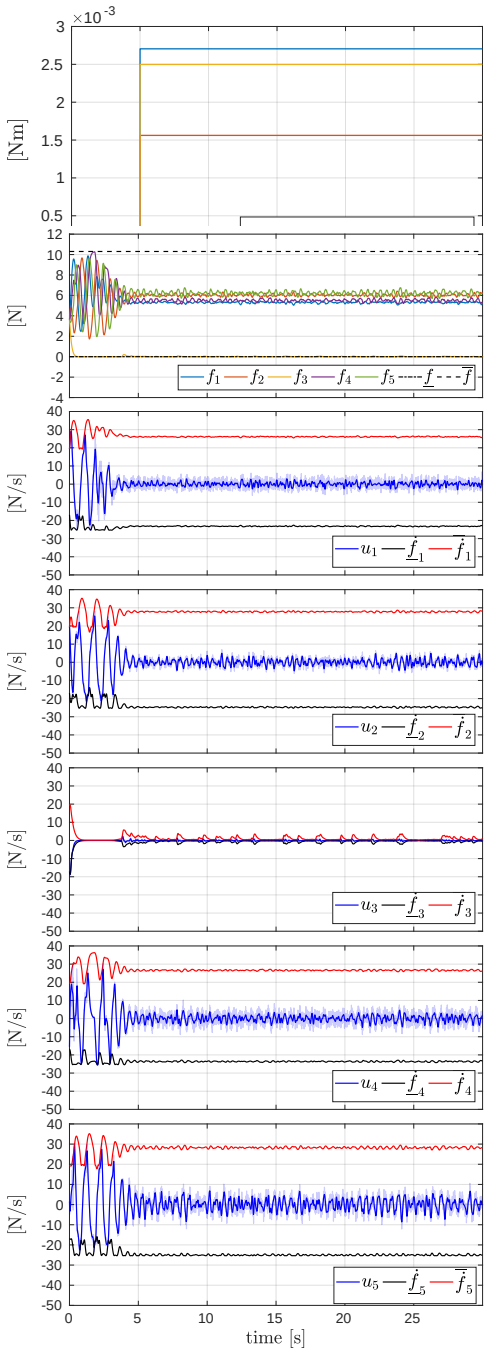

Fig. 23 Plots of the Tilt-Hex with rotor failure and $\beta = -25\,\text{deg}$ while hovering. The robot is disturbed with a constant external torque $\boldsymbol{\tau}_{\text{dist}}$, activated at $t = t_1$. From top to bottom, the disturbance torque, the actuator forces and their derivatives. In particular, all the signals remain inside the feasible region delimited by the constraints.

lates, with peaks of almost $\pm 2\,\text{cm}$ and $\pm 7.5\,\text{deg}$, around the steady-state configurations. On the other hand, for $\beta = -25\,\text{deg}$ the MRAV can fulfill the challenging goal of remaining still. This is a consequence of the fact that, for $\beta \neq 0$ the span of ${}^3\mathbf{G}_2^6$ is already 3-dimensional and so the perturbation can be annihilated while being in static hovering. In both cases, the first part of the simulation is characterized by consistent oscillations of the state components, as it is clear from the plots 1-4 of the two figures. In particular, these transients are caused by the fact that the initial robot orientation is $\boldsymbol{\eta}_0 = [0\,0\,0]^\top\,\text{deg}$, which is not attainable in steady-state for the MRAV in both configurations.

Some final remarks are detailed in order. First of all, the aforementioned claim that, in this case, the $3-rd$ actuator is almost never used is confirmed by the last plots

Preprint version,
final version at https://www.springer.com/journal/10846

of Fig. 20 and Fig. 22. Indeed, the control algorithm regulates to zero the related force component almost everywhere. In particular, during the initial transient phase, we see how the rotor velocities (and so the generated thrust forces) approach their upper bounds. Regulating the spinning rate of the $3-rd$ rotor to a value grater than zero, would cause the other components to saturate, with large chances to destabilize the platform. Secondly, the platform orientation converges (for $\beta = -25\,\mathrm{deg}$) to a certain value, as depicted in the third and in the sixth plots of Fig. 22. Note that such steady-state orientation value, which depends on α, β and on $\boldsymbol{\tau}_{\text{dist}}$, is automatically computed by the NMPC algorithm, in relation to the state and input limitations, and it is not a-priori given. This feature guarantees the optimality of the trajectory w.r.t. the robot dynamic capabilities and relieves the user from performing any explicit computations. Finally, remark that the proposed controller can achieve better results compared to the one designed in [50, 51], since the errors on the state keep bounded without diverging also in the case $\beta = 0$, despite the addition of a constant challenging disturbance which remain unknown to the NMPC algorithm. This fact highlights the potentiality of predictive controllers compared to reactive static feedback ones.

6 Conclusions

In this paper, we have presented an NMPC framework tailored to generic multi-directional thrust MRAVs with arbitrarily positioned and oriented rotors, which considers a novel and more representative model for the actuators of such systems compared to the ones often employed by other works. More in detail, the time derivatives of the propeller thrust forces are considered as the control inputs to be optimized by the predictive controller, as they are directly related to the torques applied to the motors, which constitute the lowest-level control inputs for multi-rotor systems. Thanks to the simple but effective model for the actuator dynamics that we designed by leveraging available experimental data, it is possible to indirectly take into account multiple low-level physical effects such as the ones induced by the rotor inertia, the aerodynamic drag, and other highly nonlinear hidden electrical phenomena, just by modeling the maximum force derivatives (equivalently, the maximum rotor accelerations) as a function, identifiable thanks to the proposed methodology, of the instantaneous propeller forces and the user-defined accuracy w.r.t. the set-point thrust values. During the resolution of the optimal control problem, we constrain only the inputs and the part of the model state related to the actuators to lie within the identified feasible set, avoiding the imposition of any fictitious limits in the robot orientation, angular velocity, body thrust and moment, or any other non-physical limitations. To improve its computational efficiency, the control algorithm is implemented using a state-of-the-art real time iteration scheme with partial sensitivity update method. To demonstrate its real-time capabilities, the controller has been validated with four different multi-rotor platforms, both in experiments and realistic simulations, showing its versatility and applicability to different challenging scenarios. At the best of our knowledge, this is the first time that an NMPC framework with all such features is presented and extensively validated both with under-actuated and fully-actuated aerial robots, and both with fixed and orientable propellers. Ultimately, we have provided a unified framework for the predictive control of generic multi-rotor aerial vehicles which can be particularized for the specific platform at hand just by applying the proposed identification procedure for the actuators limits, which constitutes an additional contribution of our work.

Future work includes the automatic regulation of the cost function weights, which are a fundamental part of predictive controllers, in order to reduce the tuning time and effort for the user. The use of neural networks could be envisioned. Moreover, we plan to conduct additional validation tests, e.g., including perception objectives inside the cost function, and controlling other different MDT-MRAVs. Ultimately, we aim to transfer the technology presented in the experimental validation to an outdoor MoCap-denied scenario where only on-board computational resources are used. The final goal is to be able to extend and use the presented framework to fulfill aerial physical interaction tasks.

Appendix

Allocation matrix identification

The nominal values of the entries of the allocation matrix $\mathbf{G}$ can be calculated from the system's geometrical properties, consistently with (7). However, the real physical parameters of the robot could be quite different from the ideal ones, due to mechanical inaccuracies unavoidably associated with the manufacturing and the assembly of the robot parts. This may dramatically affect the control system performances. For this reason, in this work the entries of the allocation matrix are identified from experimental data. In the following we briefly outline the used identification method, which is extensively used in the literature and very well-known from the community, so it is not considered as a contribution.

First of all, we used the nominal allocation matrix to design a simple but robust controller, applied on the platform. Accordingly, the so-obtained control system is used to track suitable *persistently exciting* 6D trajectories. To this purpose chirp signals are used, i.e., sinusoidal trajectories with increasing frequencies. While doing this, we collected the measured data $\mathbf{p}$ and $\mathbf{R}$ thanks to the MoCap system, used as ground truth. In particular, we made the assumption to be

Accepted for publication at the
Springer Journal of Intelligent & Robotic Systems, 2020

able to measure the CoM location. Then, thanks to a properly tuned post-processing of the data which mainly consisted in a constant frame-rate signal re-sampling, an anti-causal low-pass filtering and the computation of numerical derivatives, we were able to retrieve a precise-enough estimation of $\ddot{\mathbf{p}}, \boldsymbol{\omega}, \dot{\boldsymbol{\omega}}$ defined in (2). On the other hand, $\boldsymbol{\gamma}$ was reconstructed by collecting the measured spinning rates of the motors w_i and using the thrust model (5). Finally, m was directly measured and $\mathbf{J}$ estimated by a precise CAD model of the robot. At this point, we re-wrote (2) as

$$\underbrace{\begin{bmatrix} m\mathbf{R}^\top(\ddot{\mathbf{p}} + g\mathbf{e}_3) \\ \mathbf{J}\dot{\boldsymbol{\omega}} + \boldsymbol{\omega} \times \mathbf{J}\boldsymbol{\omega} \end{bmatrix}}_{:=\mathbf{y}} = \underbrace{\begin{bmatrix} f_1\mathbf{I}_3 \ \dots \ f_n\mathbf{I}_3 & \mathbf{0}_{3\times 3n} \\ \mathbf{0}_{3\times 3n} & f_1\mathbf{I}_3 \ \dots \ f_n\mathbf{I}_3 \end{bmatrix}}_{:=\mathbf{A}} \boldsymbol{\beta} \tag{41}$$

with $\mathbf{A} \in \mathbb{R}^{6\times 6n}$. In such form, the equation allows to express the vector of measurable quantities $\mathbf{y} \in \mathbb{R}^{6\times 1}$ as a linear function of a vector of parameters $\boldsymbol{\beta} \in \mathbb{R}^{6n\times 1}$, obtained re-arranging the entries of $\mathbf{G}$

$$\boldsymbol{\beta} := \begin{bmatrix} \mathbf{G}_1(:,1)^\top \ \dots \ \mathbf{G}_1(:,n)^\top \ \mathbf{G}_2(:,1)^\top \ \dots \mathbf{G}_2(:,n)^\top \end{bmatrix}^\top \tag{42}$$

Collecting a large number of measurements $p >> 6n$ and stacking them in vectorial form, we obtained

$$(\boldsymbol{\xi} = \boldsymbol{\Lambda}\boldsymbol{\beta}) := \left(\begin{bmatrix} \mathbf{y}_1 \\ \vdots \\ \mathbf{y}_p \end{bmatrix} = \begin{bmatrix} A_1 \\ \vdots \\ A_p \end{bmatrix} \boldsymbol{\beta} \right) \tag{43}$$

At this point, applying the standard least-squares identification method, the vector of parameters which minimizes the 2-norm of the error $||\boldsymbol{\Lambda}\boldsymbol{\beta} - \boldsymbol{\xi}||^2$ is obtained as

$$\hat{\boldsymbol{\beta}} = \boldsymbol{\Lambda}^\dagger \boldsymbol{\xi} \tag{44}$$

Finally, re-arranging the element of the vector $\hat{\boldsymbol{\beta}}$ using the convention of (42), we obtained the identified allocation matrix $\hat{\mathbf{G}}$ that we used in the presented experiments.

Comparing the entries of the nominal and the identified allocation matrices in the hexarotor (Tilt-Hex) case, notice that the difference between some elements is pretty consistent. This confirms that the physical parameters of the real robot can be very dissimilar from the nominal ones.

$$e_{G,\%} = 100 \left[\frac{g_{i,j} - \hat{g}_{i,j}}{g_{i,j}} \right] = \begin{bmatrix} 42 & -12 & -18 & 41 & 9 & 16 \\ 4 & 21 & 25 & 4 & 80 & 104 \\ 4 & 6 & 11 & 10 & 5 & 2 \\ 72 & 31 & 30 & 58 & 25 & 28 \\ 26 & 24 & 31 & 27 & 29 & 28 \\ 9 & 15 & 16 & 12 & 14 & 13 \end{bmatrix} \tag{45}$$

To conclude, we would like to point out that using the identified matrix in the controller instead of the nominal one allowed to consistently reduce both the position and the orientation errors in all the experiments that we performed. This happens already in hovering condition, as it is shown in the box-plots of Fig. 24.

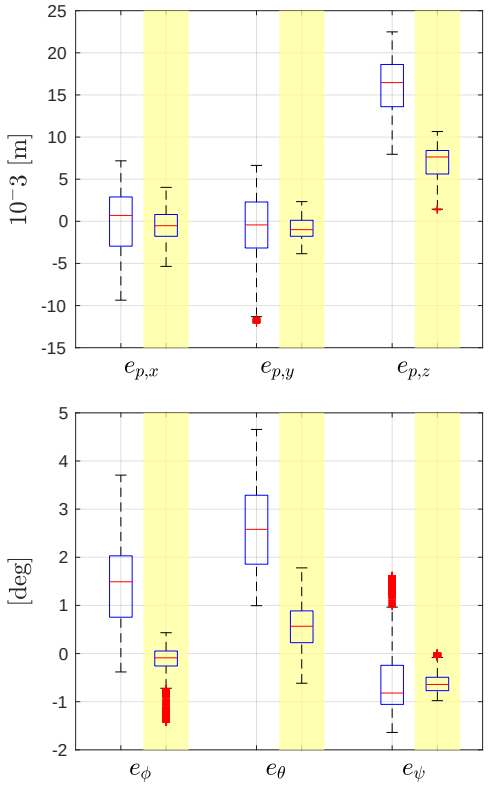

Fig. 24 Box-plots for the position error (above) and the orientation error (below) of the Tilt-Hex when hovering using the nominal and the identified allocation matrices. The results for the latter case have been highlighted with yellow bands. We can appreciate how the error mean and variance is reduced.

Acknowledgements We thank Anthony Mallet (LAAS-CNRS) for his contribution to the development of the software architecture exploited for the experiments and Yutao Chen (University of Padova – Eindhoven University) for the development of the MATMPC framework, integrated in our setup.

References

1. L. Merino, F. Caballero, J. R. Martínez-De-Dios, I. Maza, and A. Ollero, "An unmanned aircraft system for automatic forest fire monitoring and measurement," *Journal of Intelligent & Robotic Systems*, vol. 65, no. 1-4, pp. 533–548, 2012.
2. M. Ryll, G. Muscio, F. Pierri, E. Cataldi, G. Antonelli, F. Caccavale, D. Bicego, and A. Franchi, "6D interaction control with aerial robots: The flying end-effector paradigm," *The International Journal of Robotics Research*, vol. 38, no. 9, pp. 1045–1062, 2019.
3. N. Staub, D. Bicego, Q. Sablé, V. Arellano-Quintana, S. Mishra, and A. Franchi, "Towards a flying assistant paradigm: the OTHex," in *2018 IEEE Int. Conf. on Robotics and Automation*, Brisbane, Australia, May 2018, pp. 6997–7002.
4. K. Alexis, C. Papachristos, R. Siegwart, and A. Tzes, "Robust explicit model predictive flight control of unmanned rotorcrafts: Design and experimental evaluation," in *Control Conference (ECC), 2014 European*. IEEE, 2014, pp. 498–503.
5. ——, "Robust model predictive flight control of unmanned rotorcrafts," *Journal of Intelligent & Robotic Systems*, vol. 81, no. 3-4, pp. 443–469, 2016.

Preprint version,
final version at https://www.springer.com/journal/10846

References

6. T. Baca, G. Loianno, and M. Saska, "Embedded model predictive control of unmanned micro aerial vehicles," in *Methods and Models in Automation and Robotics (MMAR), 2016 21st International Conference on*. IEEE, 2016, pp. 992–997.
7. M. Bangura and R. Mahony, "Real-time model predictive control for quadrotors," *IFAC Proceedings Volumes*, vol. 47, no. 3, pp. 11 773 – 11 780, 2014, 19th IFAC World Congress.
8. P. Bouffard, A. Aswani, and C. Tomlin, "Learning-based model predictive control on a quadrotor: Onboard implementation and experimental results," in *Robotics and Automation (ICRA), 2012 IEEE International Conference on*. IEEE, 2012, pp. 279–284.
9. G. Darivianakis, K. Alexis, M. Burri, and R. Siegwart, "Hybrid predictive control for aerial robotic physical interaction towards inspection operations," in *Robotics and Automation (ICRA), 2014 IEEE International Conference on*. IEEE, 2014, pp. 53–58.
10. P. Foehn and D. Scaramuzza, "Onboard state dependent lqr for agile quadrotors," in *Robotics and Automation (ICRA), 2018 IEEE International Conference on*. IEEE, 2018.
11. M. Geisert and N. Mansard, "Trajectory generation for quadrotor based systems using numerical optimal control," in *Robotics and Automation (ICRA), 2016 IEEE International Conference on*. IEEE, 2016, pp. 2958–2964.
12. M. Hofer, M. Muehlebach, and R. D'Andrea, "Application of an approximate model predictive control scheme on an unmanned aerial vehicle," in *Robotics and Automation (ICRA), 2016 IEEE International Conference on*. IEEE, 2016, pp. 2952–2957.
13. M. Kamel, K. Alexis, M. Achtelik, and R. Siegwart, "Fast nonlinear model predictive control for multicopter attitude tracking on so (3)," in *Control Applications (CCA), 2015 IEEE Conference on*. IEEE, 2015, pp. 1160–1166.
14. 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.
15. J. A. Ligthart, P. Poksawat, L. Wang, and H. Nijmeijer, "Experimentally validated model predictive controller for a hexacopter," *IFAC-PapersOnLine*, vol. 50, no. 1, pp. 4076–4081, 2017.
16. P. Lin, S. Chen, and C. Liu, "Model predictive control-based trajectory planning for quadrotors with state and input constraints," in *Control, Automation and Systems (ICCAS), 2016 16th International Conference on*. IEEE, 2016, pp. 1618–1623.
17. C. Liu, W.-H. Chen, and J. Andrews, "Explicit non-linear model predictive control for autonomous helicopters," *Proceedings of the Institution of Mechanical Engineers, Part G: Journal of Aerospace Engineering*, vol. 226, no. 9, pp. 1171–1182, 2012.
18. Y. Liu, J. M. Montenbruck, P. Stegagno, F. Allgöwer, and A. Zell, "A robust nonlinear controller for nontrivial quadrotor maneuvers: Approach and verification," in *Intelligent Robots and Systems (IROS), 2015 IEEE/RSJ International Conference on*. IEEE/RSJ, 2015, pp. 5410–5416.
19. M. W. Mueller and R. D'Andrea, "A model predictive controller for quadrocopter state interception," in *Control Conference (ECC), 2013 European*. IEEE, 2013, pp. 1383–1389.
20. M. W. Mueller, M. Hehn, and R. D'Andrea, "A computationally efficient motion primitive for quadrocopter trajectory generation," *IEEE Transactions on Robotics*, vol. 31, no. 6, pp. 1294–1310, 2015.
21. M. Neunert, C. De Crousaz, F. Furrer, M. Kamel, F. Farshidian, R. Siegwart, and J. Buchli, "Fast nonlinear model predictive control for unified trajectory optimization and tracking," in *Robotics and Automation (ICRA), 2016 IEEE International Conference on*. IEEE, 2016, pp. 1398–1404.
22. C. Papachristos, K. Alexis, and A. Tzes, "model predictive hovering-translation control of an unmanned tri-tiltrotor," in *Robotics and Automation (ICRA), 2013 IEEE International Conference on*. IEEE, May 2013, pp. 5425–5432.
23. ——, "Dual–authority thrust–vectoring of a tri–tiltrotor employing model predictive control," *Journal of intelligent & robotic systems*, vol. 81, no. 3-4, pp. 471–504, 2016.
24. N. Michael, D. Mellinger, Q. Lindsey, and V. Kumar, "The grasp multiple micro-uav testbed," *IEEE Robot. Automat. Mag.*, vol. 17, no. 3, pp. 56–65, 2010.
25. T. Lee, M. Leoky, and N. H. McClamroch, "Geometric tracking control of a quadrotor UAV on SE(3)," in *49th IEEE Conf. on Decision and Control*, Atlanta, GA, Dec. 2010, pp. 5420–5425.
26. D. Mellinger and V. Kumar, "Minimum snap trajectory generation and control for quadrotors," in *2011 IEEE Int. Conf. on Robotics and Automation*, Shanghai, China, May. 2011, pp. 2520–2525.
27. F. Goodarzi, D. Lee, and T. Lee, "Geometric nonlinear pid control of a quadrotor uav on se (3)," in *Control Conference (ECC), 2013 European*. IEEE, 2013, pp. 3845–3850.
28. S. Dai, T. Lee, and D. S. Bernstein, "Adaptive control of a quadrotor uav transporting a cable-suspended load with unknown mass," in *Decision and Control (CDC), 2014 IEEE 53rd Annual Conference on*. IEEE, 2014, pp. 6149–6154.
29. S. Bouabdallah and R. Siegwart, "Backstepping and sliding-mode techniques applied to an indoor micro quadrotor," in *2005 IEEE Int. Conf. on Robotics and Automation*, May 2005, pp. 2247–2252.
30. M.-D. Hua, T. Hamel, P. Morin, and C. Samson, "Introduction to feedback control of underactuated VTOL vehicles: A review of basic control design ideas and principles," *IEEE Control Systems Magazine*, vol. 33, no. 1, pp. 61–75, 2013.
31. A. Franchi, R. Carli, D. Bicego, and M. Ryll, "Full-pose tracking control for aerial robotic systems with laterally-bounded input force," *IEEE Trans. on Robotics*, vol. 34, no. 2, pp. 534–541, 2018.
32. S.-J. Chung, A. A. Paranjape, P. Dames, S. Shen, and V. Kumar, "A survey on aerial swarm robotics," *IEEE Transactions on Robotics*, vol. 34, no. 4, pp. 837–855, 2018.
33. M. Ryll, H. H. Bülthoff, and P. Robuffo Giordano, "A novel overactuated quadrotor unmanned aerial vehicle: modeling, control, and experimental validation," *IEEE Trans. on Control Systems Technology*, vol. 23, no. 2, pp. 540–556, 2015.
34. W. Khan and M. Nahon, "Toward an accurate physics-based uav thruster model," *IEEE/ASME Transactions on Mechatronics*, vol. 18, no. 4, pp. 1269–1279, 2013.
35. V. Arellano-Quintana, E. Merchan-Cruz, and A. Franchi, "A novel experimental model and a drag-optimal allocation method for variable-pitch propellers in multirotors," *IEEE Access*, vol. 6, no. 1, pp. 68 155–68 168, 2018.
36. F. Morbidi, D. Bicego, M. Ryll, and A. Franchi, "Energy-efficient trajectory generation for a hexarotor with dual-tilting propellers," in *2018 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, Madrid, Spain, Oct. 2018.
37. D. Brescianini and R. D'Andrea, "Design, modeling and control of an omni-directional aerial vehicle," in *2016 IEEE Int. Conf. on Robotics and Automation*, Stockholm, Sweden, May 2016, pp. 3261–3266.
38. S. Park, J. J. Her, J. Kim, and D. Lee, "Design, modeling and control of omni-directional aerial robot," in *2016 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, Daejeon, South Korea, 2016, pp. 1570–1575.
39. M. Tognon and A. Franchi, "Omnidirectional aerial vehicles with unidirectional thrusters: Theory, optimal design, and control," *IEEE Robotics and Automation Letters*, vol. 3, no. 3, pp. 2277–2282, 2018.
40. R. Mahony, V. Kumar, and P. Corke, "Multirotor aerial vehicles," *IEEE Robotics and Automation magazine*, vol. 20, no. 32, 2012.
41. A. Bemporad, C. A. Pascucci, and C. Rocchi, "Hierarchical and hybrid model predictive control of quadcopter air vehicles," *IFAC Proceedings Volumes*, vol. 42, no. 17, pp. 14–19, 2009.
42. A. Franchi and A. Mallet, "Adaptive closed-loop speed control of BLDC motors with applications to multi-rotor aerial vehicles," in *2017 IEEE Int. Conf. on Robotics and Automation*, Singapore, May 2017, pp. 5203–5208.
43. D. Mayne, J. Rawlings, C. Rao, and P. Scokaert, "Constrained model predictive control: Stability and optimality," *Automatica*, vol. 36, no. 6, pp. 789 – 814, 2000.

References

44. M. Alamir, "Stability proof for nonlinear mpc design using monotonically increasing weighting profiles without terminal constraints," *Automatica*, vol. 87, pp. 455–459, 2018.
45. A. Alessandretti, A. P. Aguiar, and C. N. Jones, "Trajectory-tracking and path-following controllers for constrained underactuated vehicles using model predictive control," in *2013 European Control Conference (ECC)*, 2013, pp. 1371–1376.
46. L. Grüne, J. Pannek, M. Seehafer, and K. Worthmann, "Analysis of unconstrained nonlinear mpc schemes with time varying control horizon," *SIAM Journal on Control and Optimization*, vol. 48, no. 8, pp. 4938–4962, 2010.
47. M. Alamir and G. Bornard, "Stability of a truncated infinite constrained receding horizon scheme: the general discrete nonlinear case," *Automatica*, vol. 31, no. 9, pp. 1353 – 1356, 1995.
48. M. W. Mehrez, K. Worthmann, G. K. Mann, R. G. Gosine, and T. Faulwasser, "Predictive path following of mobile robots without terminal stabilizing constraints," *IFAC-PapersOnLine*, vol. 50, no. 1, pp. 9852 – 9857, 2017.
49. M. Faessler, A. Franchi, and D. Scaramuzza, "Differential flatness of quadrotor dynamics subject to rotor drag for accurate tracking of high-speed trajectories," *IEEE Robotics and Automation Letters*, vol. 3, no. 2, pp. 620–626, 2018.
50. G. Michieletto, M. Ryll, and A. Franchi, "Control of statically hoverable multi-rotor aerial vehicles and application to rotor-failure robustness for hexarotors," in *2017 IEEE Int. Conf. on Robotics and Automation*, Singapore, May 2017, pp. 2747–2752.
51. ——, "Fundamental actuation properties of multi-rotors: Force-moment decoupling and fail-safe robustness," *IEEE Trans. on Robotics*, vol. 34, no. 3, pp. 702–715, 2018.
52. B. R. Knudsen, A. Alessandretti, and C. N. Jones, "Sensor fault tolerance in output feedback nonlinear model predictive control," in *2016 3rd Conference on Control and Fault-Tolerant Systems (SysTol)*. Ieee, 2016, pp. 711–716.
53. M. Diehl, H. G. Bock, J. P. Schlöder, R. Findeisen, Z. Nagy, and F. Allgöwer, "Real-time optimization and nonlinear model predictive control of processes governed by differential-algebraic equations," *Journal of Process Control*, vol. 12, no. 4, pp. 577–585, 2002.
54. H. G. Bock and K.-J. Plitt, "A multiple shooting algorithm for direct solution of optimal control problems," in *Proceedings of the IFAC World Congress*, 1984.
55. Y. Chen, D. Cuccato, M. Bruschetta, and A. Beghi, "A fast nonlinear model predictive control strategy for real-time motion control of mechanical systems," in *Advanced Intelligent Mechatronics (AIM), 2017 IEEE International Conference on*. IEEE, 2017, pp. 1780–1785.
56. J. Andersson, "A general-purpose software framework for dynamic optimization," Ph.D. dissertation, PhD thesis, Arenberg Doctoral School, KU Leuven, Department of Electrical Engineering (ESAT/SCD) and Optimization in Engineering Center, Kasteelpark Arenberg 10, 3001-Heverlee, Belgium, 2013.
57. H. Ferreau, C. Kirches, A. Potschka, H. Bock, and M. Diehl, "qpOASES: A parametric active-set algorithm for quadratic programming," *Mathematical Programming Computation*, vol. 6, no. 4, pp. 327–363, 2014.
58. Y. Chen, M. Bruschetta, E. Picotti, and A. Beghi, "Matmpc-a matlab based toolbox for real-time nonlinear model predictive control," in *2019 18th European Control Conference (ECC)*. IEEE, 2019, pp. 3365–3370.
59. D. Falanga, P. Foehn, P. Lu, and D. Scaramuzza, "Pampc: Perception-aware model predictive control for quadrotors," in *Intelligent Robots and Systems (IROS), 2018 IEEE/RSJ International Conference on*. IEEE/RSJ, 2018.
60. M. Ryll, D. Bicego, and A. Franchi, "Modeling and control of FAST-Hex: a fully-actuated by synchronized-tilting hexarotor," in *2016 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, Daejeon, South Korea, Oct. 2016, pp. 1689–1694.
61. M. Achtelik, K.-M. Doth, D. Gurdan, and J. Stumpf, "Design of a multi rotor mav with regard to efficiency, dynamics and redundancy," in *AIAA Guidance, Navigation, and Control Conference*, 2012, p. 4779.
62. J. I. Giribet, R. S. Sanchez-Pena, and A. S. Ghersin, "Analysis and design of a tilted rotor hexacopter for fault tolerance," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 52, no. 4, pp. 1555–1567, 2016.