IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.

QP-Based Inner-Loop Control for Constraint-Safe and Robust Trajectory Tracking for Aerial Robots

Lorenzo Balandi[1], Paolo Robuffo Giordano[1], and Marco Tognon[1]

***Abstract*—Accurate trajectory tracking is crucial in aerial robotics. Optimal control methods such as Nonlinear Model Predictive Control (NMPC) are able to track trajectories exploiting the full nonlinear dynamics while respecting constraints. However, the NMPC model-based nature makes it sensitive to mismatches among nominal and real models. A common workaround to mitigate the effects of model uncertainties is to implement an inner-loop controller which robustifies the NMPC outer-loop. However, this inner-loop is usually based on purely feedback-based controllers such as PID or Incremental Nonlinear Dynamic Inversion (INDI), which do not allow to consider any constraint (such as limited actuation) or optimization criteria. In contrast, in this work we propose an optimization-based inner-loop controller inspired by Time Delay Control (TDC), that, thanks to a Quadratic Program (QP) formulation, is able to respect constrains and can thus preserve stability in presence of input saturation and model mismatches. Furthermore, thanks to the use of acceleration feedback, the knowledge of inertial parameters is not required by the proposed inner-loop which therefore makes it even more robust against model uncertainties. The overall architecture is validated on a fully-actuated hexarotor under model mismatches and aggressive trajectories. The experiments clearly show that our QP-based inner-loop improves the NMPC tracking performance while preserving the stability in conditions where a non-optimal (and more classical) inner-loop controllers would fail.**

***Index Terms*—Aerial Systems: Mechanics and Control, Motion Control, Optimization and Optimal Control**

I. Introduction

PRECISE trajectory tracking controllers are crucial for Micro Aerial Vehicles (MAVs) to execute both contactless and contact-based tasks effectively [1], as they must reject external disturbances (e.g., wind) and be robust against unmodeled dynamics (e.g., aerodynamics). Over the last decade, many control strategies have been proposed to deal with the inherently unstable and nonlinear nature of MAVs, and in particular of quadrotors. For instance, a very popular choice is to rely on geometric controllers that can explicitly deal with the MAVs nonlinearities [2]. The differential flatness property has also been exploited for the tracking of aggressive trajectories [3], [4] with good results also on real experiments on quadrotors with onboard computational capabilities.

Fig. 1: Fully-actuated hexarotor used in the experiments with reference frames.

A drawback is that the control law is not able to take into account performance criteria (e.g., energy consumption) or constraints (e.g., limited actuation); moreover, the control input is only based on the current state of the system and cannot reason about longer-term consequences of local actions (e.g., avoiding future collisions). Model Predictive Control (MPC) is a popular optimization-based control strategy that can instead explicitly handle constraints and/or optimality, and which can reason over a longer time horizon compared to a purely reactive feedback. MPC and NMPC have been successfully applied to trajectory tracking for under-actuated, fully-actuated and omnidirectional MAVs [4]–[7]. This control approach exploits a dynamic model of the system to predict its future behavior and optimize the control actions over a time horizon. This makes it possible, for instance, to effectively track unfeasible reference trajectories while respecting input and state constraints at the cost, however, of a higher computational load with respect to (wrt) simpler feedback-based controllers.

The above mentioned controls, including MPC, depend on an accurate knowledge of the robot model. To mitigate the effects of model uncertainties, robust control schemes, such as Sliding Mode [8] and $\mathcal{H}_\infty$ [9] have been proposed in the literature. To robustify MPC, an inner-loop controller is often employed with the MPC acting as outer-loop. For this purpose, a common choice is the INDI [10], [11]: this sensor-based approach enables high-performance tracking through direct acceleration measurement or estimation and is derived

Manuscript received: December, 23, 2024; Revised February, 25, 2025 ; Accepted April, 27, 2025.

This paper was recommended for publication by Editor Giuseppe Loianno upon evaluation of the Associate Editor and Reviewers' comments.

This work was supported by the project ANR-23-CE33-0013 "FlyHandyBot". Experiments presented in this paper were carried out thanks to a platform of the Robotex 2.0 French research infrastructure.

[1] Univ Rennes, CNRS, Inria, IRISA, Rennes, 35042, France, lorenzo.balandi@inria.fr, marco.tognon@inria.fr, prg@irisa.fr.

Digital Object Identifier (DOI): see top of this page.

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.

from applying Nonlinear Dynamic Inversion (NDI) to an incremental system (equivalent formulations are available in the literature [10]). An alternative inner-loop controller for NMPC is $\mathcal{L}_1$ adaptive control, successfully applied in [12], where the authors show that the inner-loop is able to deal with modeling errors similarly to an integral action, unlike the INDI inner-loop in [4]. However, note that different INDI formulations that more closely mirror the original one can provide the same effect [3] .

Another control method similar to INDI is TDC [13], originally proposed as a possible input/output linearization method for uncertain nonlinear systems. More recent derivations of TDC can be found in [14]–[16]. The basic idea is to use sensors to measure the unknown dynamics and compensate it in an incremental fashion. TDC has been applied to robotic manipulators in [15], [16], where a discrete-time incremental system is derived and used as dynamic model for MPC. While INDI still requires a (limited) knowledge of the model, TDC is potentially *model-free* thanks to the approximation of unknown dynamics from acceleration measurements. Unlike ground manipulators [15], [16], which control joint torques directly, MAVs control propeller speeds rather than the body wrench. Therefore, in the MAVs case an approach based on TDC cannot be fully *model-free*.

All mentioned inner-loop controllers are suitable choices for increasing the robustness of the MPC outer-loop against model uncertainties thanks to their *model-free* nature (though to different extents). However, unlike MPC or other optimization-based controls, they do not offer the possibility of explicitly considering input limits or other constraints. Therefore their use as inner-loop controllers can still lead to a constraint violation and possible instability in case of model mismatches or other disturbances when used in conjunction with an outer MPC loop, event if no additional mismatches are added. In this context, the goal of this paper is to address these issues by proposing a novel high-frequency inner-loop controller that embeds TDC into a QP problem, building on the ideas introduced in [15], [16]. This results in two main benefits: (i) the robustness of the outer MPC loop against model uncertainties is still enhanced thanks to the presence of the TDC inner-loop and (ii) actuation constraints are strictly met also at the inner-loop level thanks to the QP optimization formulation, thus addressing the aforementioned shortcoming of previous inner-loop controllers.

The proposed QP-based inner-loop, referred to as *TDC-QP*, significantly reduces tracking errors without requiring precise knowledge of the MAV model parameters, showing a behavior comparable to [12]. At the same time, it ensures closed-loop stability even in the presence of input saturation—conditions where previously mentioned inner-loops would fail. These claims are further validated through the experimental results presented in Sec. V, which involve substantial model mismatches and aggressive reference trajectories that induce input saturation, demonstrating the robust performance of TDC-QP when the system operates at its limits.

The rest of the paper is structured as follows: Sec. II presents the dynamic model for the hexarotor used as case study in this work, while Sec. III illustrates the NMPC formulation. Sec. IV introduces the main contribution of the paper, that is, the derivation of an incremental system based on TDC and its embedding in a QP, as well as its practical implementation as inner-loop for NMPC and a discussion on its relation with INDI. The experimental validation is discussed in Sec. V and Sec. VI draws the conclusions of this work.

II. System Modeling

We model a generic MAV with $n_p \geq 6$ propellers as a 6-DoF rigid body. To describe its motion, we define a *robot frame* $\mathcal{F}_\mathcal{R} = \{O_R, \boldsymbol{x}_R, \boldsymbol{y}_R, \boldsymbol{z}_R\}$ attached to the vehicle Center of Mass (CoM), and a *world frame* $\mathcal{F}_\mathcal{W} = \{O_W, \boldsymbol{x}_W, \boldsymbol{y}_W, \boldsymbol{z}_W\}$, with arbitrarily placed origin O_W, and $\boldsymbol{z}_W$ opposing gravity vector. Reference frames are depicted in Fig. 1. The position of O_R with respect to $\mathcal{F}_\mathcal{W}$ is denoted as $\boldsymbol{p} \in \mathbb{R}^3$, while the orientation (attitude) of $\mathcal{F}_\mathcal{R}$ wrt $\mathcal{F}_\mathcal{W}$ is represented by the rotation matrix $\boldsymbol{R} = [\boldsymbol{x}_R\ \boldsymbol{y}_R\ \boldsymbol{z}_R] \in \mathrm{SO}(3)$, parametrized by the unit quaternion $\boldsymbol{q} = [q_w\ q_x\ q_y\ q_z]^\top \in \mathbb{S}(3)$. The angular velocity of $\mathcal{F}_\mathcal{R}$ wrt $\mathcal{F}_\mathcal{W}$, expressed in $\mathcal{F}_\mathcal{R}$, is denoted as $\boldsymbol{\omega} \in \mathbb{R}^3$, $\boldsymbol{v} \in \mathbb{R}^3$ is the linear velocity in $\mathcal{F}_\mathcal{W}$.

Following the Newton-Euler formalism, expressing the translational dynamics in world frame and the rotational dynamics in robot frame, one can write the MAV model as

$$\dot{\boldsymbol{p}} = \boldsymbol{v}, \quad \dot{\boldsymbol{q}} = \frac{1}{2}\boldsymbol{q} \otimes \begin{bmatrix} 0 & \boldsymbol{\omega}^\top \end{bmatrix}^\top \tag{1a}$$

$$\begin{bmatrix} m\boldsymbol{I}_3 & \boldsymbol{0}_3 \\ \boldsymbol{0}_3 & \boldsymbol{J} \end{bmatrix} \begin{bmatrix} \dot{\boldsymbol{v}} \\ \dot{\boldsymbol{\omega}} \end{bmatrix} = \begin{bmatrix} -mg\boldsymbol{z}_W \\ -\boldsymbol{\omega} \times \boldsymbol{J}\boldsymbol{\omega} \end{bmatrix} + c_f \begin{bmatrix} \boldsymbol{R} & \boldsymbol{0}_3 \\ \boldsymbol{0}_3 & \boldsymbol{I}_3 \end{bmatrix} \boldsymbol{G}\boldsymbol{w}^{\circ 2} \tag{1b}$$

where $\otimes$ is the quaternion product, m is the robot mass, $-g\boldsymbol{z}_W$ is the gravity vector, $\boldsymbol{J}$ is the constant inertia matrix of the robot, c_f is the thrust coefficient, and $\boldsymbol{I}_3$ and $\boldsymbol{0}_3$ are the 3×3 identity and zero matrices respectively. The term multiplying the linear/angular accelerations in (1b) is called generalized inertia matrix and will be denoted as $\boldsymbol{M}$ in the following derivations. Vector $\boldsymbol{w}^{\circ 2}$ is the component-wise square of the rotor speeds $\boldsymbol{w} \in \mathbb{R}^{n_p}$ and will be taken as *control input*. As for the quantity $\boldsymbol{G} \in \mathbb{R}^{6\times n_p}$, it represents the allocation matrix computed as in [6]. By denoting $\boldsymbol{\gamma} = c_f \boldsymbol{w}^{\circ 2}$ as the vector of propeller thrusts, the quantity $\boldsymbol{G}\boldsymbol{\gamma} = [\boldsymbol{f}^\top \boldsymbol{\tau}^\top]^\top \in \mathbb{R}^6$ is the body wrench. Note that for fully-actuated aerial vehicles, like the one considered in this work, $\boldsymbol{G}$ is full rank, while for standard quadrotors its rank is limited to 4.

The value of the various parameters in (1) (center of mass, geometrical quantities, thrust coefficient, etc) is in general only approximately known. Therefore, we introduce a *nominal* dynamic model

$$\begin{bmatrix} \hat{m}\boldsymbol{I}_3 & \boldsymbol{0}_3 \\ \boldsymbol{0}_3 & \hat{\boldsymbol{J}} \end{bmatrix} \begin{bmatrix} \dot{\boldsymbol{v}} \\ \dot{\boldsymbol{\omega}} \end{bmatrix} = \begin{bmatrix} -\hat{m}g\boldsymbol{z}_W \\ -\boldsymbol{\omega} \times \hat{\boldsymbol{J}}\boldsymbol{\omega} \end{bmatrix} + \hat{c}_f \begin{bmatrix} \boldsymbol{R} & \boldsymbol{0}_3 \\ \boldsymbol{0}_3 & \boldsymbol{I}_3 \end{bmatrix} \hat{\boldsymbol{G}}\boldsymbol{w}^{\circ 2} \tag{2}$$

where the $\hat{\cdot}$ indicates the approximated knowledge of uncertain parameters due to measurement or estimation errors. The estimated generalized inertia matrix will be denoted as $\hat{\boldsymbol{M}}$. This nominal model will be relevant in the following developments.

A. Problem Statement and Control Framework Overview

The goal of the control framework discussed in the following is to track a desired state trajectory for the robot under

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.

model uncertainties. To this end, an NMPC is proposed based on the *nominal* model (2). Given the mismatch between real and nominal dynamical models (1)–(2), the NMPC will present considerable tracking errors due to its model-based nature. A TDC-based controller is thus developed for fully-actuated MAVs and integrated in the control scheme as inner-loop for the NMPC. This additional loop computes control inputs for the MAV given states and inputs from the NMPC and reduces the tracking error thanks to acceleration feedback that mitigates the effects of model uncertainties. However, the TDC inner-loop is not able to handle constraints and, thus, even if the inputs generated by the NMPC would respect actuation limits the TDC action can still incur into input saturation and introduce instability in presence of model mismatches. To solve this issue, we then embed the TDC inner-loop into a QP which guarantees the satisfaction of the same input constraints as in the NMPC outer-loop.

III. Nonlinear Model Predictive Controller

We rely on a NMPC for solving the trajectory tracking task. NMPC iteratively solves a constrained Optimal Control Problem (OCP) and computes optimal states and inputs over a (finite) prediction horizon by forward integrating the MAV nominal model (2). In our case, the state and input vectors are $\boldsymbol{x} = \left[\boldsymbol{p}^\top \boldsymbol{q}^\top \boldsymbol{v}^\top \boldsymbol{\omega}^\top\right]^\top \in \mathbb{R}^3 \times \mathbb{S}(3) \times \mathbb{R}^6$ and $\boldsymbol{u} = \boldsymbol{w}^{\circ 2} \in \mathbb{R}^{n_p}$, respectively. The control objective is to track as close as possible a desired time-varying state trajectory $\boldsymbol{x}_d(t) = \left[\boldsymbol{p}_d(t)^\top \boldsymbol{q}_d(t)^\top \boldsymbol{v}_d(t)^\top \boldsymbol{\omega}_d(t)^\top\right]^\top$, which may not be dynamically feasible considering actuation constraints, and possibly a reference input $\boldsymbol{u}_d(t)$ computed by a trajectory planner or set to a constant value (e.g., zero or hovering value).

In order to numerically solve the OCP, states and inputs are typically discretized using N equally spaced intervals with discretization step dt and denoted with subscript $\cdot_k$. The references are also sampled and fed to the NMPC accordingly, obtaining the following Nonlinear Program (NLP):

$$\min_{\substack{\boldsymbol{x}_0,\ \dots,\ \boldsymbol{x}_N,\\ \boldsymbol{u}_0,\ \dots,\ \boldsymbol{u}_{N-1}}} \sum_{k=0}^{N-1} \left(||\boldsymbol{x}_k - \boldsymbol{x}_{k,d}||^2_{2,\boldsymbol{Q}} + ||\boldsymbol{u}_k - \boldsymbol{u}_{k,d}||^2_{2,\boldsymbol{U}}\right) \quad (3a)$$
$$+ ||\boldsymbol{x}_N - \boldsymbol{x}_{N,d}||^2_{2,\boldsymbol{Q}_N}$$
$$\text{subject to} \quad \boldsymbol{x}_{k+1} = \hat{\boldsymbol{f}}(\boldsymbol{x}_k, \boldsymbol{u}_k) \quad k \in [0, N-1] \quad (3b)$$
$$\boldsymbol{x}_0 = \hat{\boldsymbol{x}} \quad (3c)$$
$$\boldsymbol{u}_{min} \leq \boldsymbol{u}_k \leq \boldsymbol{u}_{max} \quad k \in [0, N-1], \quad (3d)$$

where $\boldsymbol{Q} = diag(\boldsymbol{Q_p}, \boldsymbol{Q_q}, \boldsymbol{Q_v}, \boldsymbol{Q_\omega})$, $\boldsymbol{Q}_\star \in \mathbb{R}^{3\times 3}$ and $\boldsymbol{U} \in \mathbb{R}^{n_p \times n_p}$ are positive definite diagonal weight matrices, $\boldsymbol{Q}_N = \boldsymbol{Q}$ (we are equally interested in all nodes), constraint (3b) is the discretized version of the nominal dynamics (2), $\hat{\boldsymbol{x}}$ is the current robot state (assumed available from state estimation), $\boldsymbol{u}_{min}$ and $\boldsymbol{u}_{max}$ are the minimum and maximum squared rotors velocities, respectively. For the attitude error, we use the vectorial part of the quaternion error with the following abuse of notation as in [4]:

$$\boldsymbol{q}^d_{err} = \boldsymbol{q} - \boldsymbol{q}_d = (\boldsymbol{q} \otimes \boldsymbol{q}_d^{-1})_{x,y,z}. \quad (4)$$

IV. Inner-loop Controller

Here we derive a TDC for fully-actuated MAVs and discuss its similarities with INDI. We then show how to embed the TDC in a QP with actuation constraints and integrate the inner-loop controller with the NMPC outer-loop. Finally, we discuss practical considerations for implementation.

A. TDC for Fully-actuated MAVs

We report here the application of TDC to fully-actuated MAVs in order to obtain an incremental system where uncertain dynamics terms are approximated with sensors measurements, following the procedure explained in [15], [16]. We demonstrate that, in the case of MAVs, knowledge of inertial parameters is unnecessary; however, the approach still depends on the allocation matrix, specifically on certain geometrical parameters and propeller coefficients.

The MAV dynamics (1) can be rewritten as a nonlinear system in control affine form

$$\dot{\boldsymbol{x}}_1 = \boldsymbol{h}(\boldsymbol{x}_1, \boldsymbol{x}_2) \quad (5a)$$
$$\dot{\boldsymbol{x}}_2 = \boldsymbol{f}(\boldsymbol{x}) + \boldsymbol{F}(\boldsymbol{x})\boldsymbol{u}, \quad (5b)$$

where $\boldsymbol{x}_1 = [\boldsymbol{p}^\top \ \boldsymbol{q}^\top]^\top \in \mathbb{R}^3 \times \mathbb{S}(3)$, $\boldsymbol{x}_2 = [\boldsymbol{v}^\top \ \boldsymbol{\omega}^\top]^\top \in \mathbb{R}^6$, $\boldsymbol{u} = \boldsymbol{w}^{\circ 2} \in \mathbb{R}^{n_p}$, $\boldsymbol{x} = \left[\boldsymbol{x}_1^\top \ \boldsymbol{x}_2^\top\right]^\top$ and $\boldsymbol{h}(\cdot)$ is a function expressing the non-linearity of the quaternion kinematics (1a). Moreover,

$$\boldsymbol{f}(\boldsymbol{x}) = \boldsymbol{M}^{-1} \begin{bmatrix} -mg\boldsymbol{z}_W \\ -\boldsymbol{\omega} \times \boldsymbol{J}\boldsymbol{\omega} \end{bmatrix}, \ \boldsymbol{F}(\boldsymbol{x}) = \boldsymbol{M}^{-1} c_f \bar{\boldsymbol{R}} \boldsymbol{G}, \quad (6)$$

with $\bar{\boldsymbol{R}} = \mathrm{diag}\,(\boldsymbol{R}, \boldsymbol{I}_3)$. Introducing matrix

$$\tilde{\boldsymbol{G}} = \bar{\boldsymbol{G}}\bar{\boldsymbol{R}}\hat{\boldsymbol{G}} \ \in \mathbb{R}^{6\times n_p} \quad (7)$$

where $\bar{\boldsymbol{G}} \in \mathbb{R}^{6\times 6}$ is a constant diagonal positive-definite gain matrix and $\hat{\boldsymbol{G}}$ the best approximation of the allocation matrix, (5b) can be rewritten as

$$\tilde{\boldsymbol{G}}^+ \dot{\boldsymbol{x}}_2 = \boldsymbol{H}(\boldsymbol{x}, \dot{\boldsymbol{x}}) + \boldsymbol{u}, \quad (8)$$

with $\boldsymbol{H}(\boldsymbol{x}, \dot{\boldsymbol{x}}) = (\tilde{\boldsymbol{G}}^+ - \boldsymbol{F}^+(\boldsymbol{x}))\dot{\boldsymbol{x}}_2 + \boldsymbol{F}^+(\boldsymbol{x})\boldsymbol{f}(\boldsymbol{x}) \in \mathbb{R}^{n_p}$. If $n_p > 6$, $\tilde{\boldsymbol{G}}$ and $\boldsymbol{F}$ are not square and their Moore-Penrose pseudoinverse $\tilde{\boldsymbol{G}}^+$, $\boldsymbol{F}^+ \in \mathbb{R}^{n_p \times 6}$ must be used.

The purpose of matrix $\bar{\boldsymbol{G}}$ is not only to provide a tunable gain for the body wrench, but also to approximate the thrust coefficient and inverse of generalized inertia matrix. The quantity $\boldsymbol{H}(\boldsymbol{x}, \dot{\boldsymbol{x}})$ includes the terms $\boldsymbol{f}(\boldsymbol{x})$ and $\boldsymbol{F}(\boldsymbol{x})$ which are assumed unknown or uncertain. According to [15], [16], these terms can be approximated using acceleration measurements. Considering a sufficiently small time interval Δt, the value of $\boldsymbol{H}(\boldsymbol{x}, \dot{\boldsymbol{x}})$ at time t can be considered close to its sampled value at time $t - \Delta t$. Therefore, the uncertain terms in (8) can be approximated using sensors, state estimation algorithms and input measurements, obtaining the Time Delay Estimation (TDE) of $\boldsymbol{H}(\boldsymbol{x}, \dot{\boldsymbol{x}})$ as

$$\hat{\boldsymbol{H}}(\boldsymbol{x}, \dot{\boldsymbol{x}}) = \tilde{\boldsymbol{G}}^+ \dot{\boldsymbol{x}}_{2,0} - \boldsymbol{u}_0, \quad (9)$$

where $\dot{\boldsymbol{x}}_{2,0} = \dot{\boldsymbol{x}}_2(t - \Delta t)$ and $\boldsymbol{u}_0 = \boldsymbol{u}(t - \Delta t)$. Note that (9) uses the most recent sampled values of accelerations $\dot{\boldsymbol{x}}_2$ and inputs $\boldsymbol{u}$ to approximate the unknown dynamics, thus

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.

making this method a *sensor-based* approach. Following the TDC literature, Δt is the sensors sampling time ($1\,\mathrm{ms}$ in our case.)

By replacing $\boldsymbol{H}(\boldsymbol{x}, \dot{\boldsymbol{x}})$ in (8) with its TDE computed in (9), we obtain the incremental dynamics

$$\dot{\boldsymbol{x}}_2 = \dot{\boldsymbol{x}}_{2,0} + \tilde{\boldsymbol{G}}(\boldsymbol{u} - \boldsymbol{u}_0) = \dot{\boldsymbol{x}}_{2,0} + \tilde{\boldsymbol{G}}\Delta\boldsymbol{u}, \tag{10}$$

which will be used in the QP formulation. Vector $\Delta\boldsymbol{u}$ is the incremental input. Note that (10) does not depend on the generalized mass matrix $\boldsymbol{M}$ anymore, but there remains a dependency on the allocation matrix $\hat{\boldsymbol{G}}$ through $\tilde{\boldsymbol{G}}$. TDC applied to MAVs is therefore not completely model-free because of the dependency on geometrical quantities and the ratio between drag and thrust through $\hat{\boldsymbol{G}}$ (see [6], Section 2.1). As previously mentioned, this limitation is due to the fact that the measured variables are linear acceleration and angular velocity whereas the controlled ones are the motor rotational speeds instead of the body-frame wrench.

Furthermore, in (10) we omitted the TDE error, see [15] and [16] for a complete discussion. Gain $\bar{\boldsymbol{G}}$ is tuned in order to satisfy a condition which guarantees the TDE error boundedness, as discussed in [13] Sec. III, [15] Sec. II-B and [16] Sec. III-A. As general suggestions to select this parameter, consider that large $\bar{\boldsymbol{G}}$ results in large TDE errors, while small $\bar{\boldsymbol{G}}$ causes noisy responses. To tune $\bar{\boldsymbol{G}}$, we can start with a sufficiently large positive value that guarantees stability and then decreases it until the closed loop response starts to become noisy. Indeed, this manual tuning procedure is in practice limited by the sensors noise level.

Note that our derivation applied to the MAV case slightly misuses the term TDC, because $\tilde{\boldsymbol{G}}$ depends on the state via the rotation matrix in $\bar{\boldsymbol{R}}$. However, since a rotation matrix has unitary norm and only changes the direction of a vector and not its magnitude, we still consider it a TDC. The key property of fixed, tunable gain remains by assuming no uncertainty in the attitude provided by the state estimator.

By solving (10) for the input $\boldsymbol{u}$, one obtains

$$\boldsymbol{u} = \boldsymbol{u}_0 + \tilde{\boldsymbol{G}}^{+}(\dot{\boldsymbol{x}}_2 - \dot{\boldsymbol{x}}_{2,0}), \tag{11}$$

where the acceleration $\dot{\boldsymbol{x}}_2$ includes the reference acceleration and a stabilizing PD action [13], [11], [14], [15] as follows

$$\dot{\boldsymbol{x}}_2 = \boldsymbol{\nu} = \dot{\boldsymbol{x}}_2^{ref} + \begin{bmatrix} \boldsymbol{K_p}(\boldsymbol{p}_{ref} - \boldsymbol{p}) + \boldsymbol{K_v}(\boldsymbol{v}_{ref} - \boldsymbol{v}) \\ \boldsymbol{K_q}\boldsymbol{q}_{err}^{ref} + \boldsymbol{K_\omega}(\boldsymbol{\omega}_{ref} - \boldsymbol{\omega}) \end{bmatrix} \tag{12}$$

with $\boldsymbol{K}_\star$ being suitable constant diagonal gain matrices and $\boldsymbol{q}_{err}^{ref}$ similar to (4). Note that this PD action is crucial to compensate model mismatches [17], as shown by experimental results. (12) allows tracking a *reference* state computed interpolating optimal states and inputs from NMPC (3), as detailed in Sec. IV-D. Control law (11) can be used as inner-loop controller to enhance the NMPC tracking capabilities and, in particular, increase its robustness against model uncertainties.

B. Relations with INDI

The control law in (11) closely resembles the INDI law, which, when applied to the fully-actuated MAV case, can be expressed as:

$$\boldsymbol{u} = \boldsymbol{u}_0 + \tilde{\boldsymbol{G}}(\boldsymbol{x}_0)^{+}(\dot{\boldsymbol{x}}_2 - \dot{\boldsymbol{x}}_{2,0}), \tag{13}$$

where $\tilde{\boldsymbol{G}}(\boldsymbol{x}_0) = \bar{\boldsymbol{G}}(\boldsymbol{x}_0)\bar{\boldsymbol{R}}\hat{\boldsymbol{G}}$ and $\dot{\boldsymbol{x}}_2 = \boldsymbol{\nu}$. A nice and detailed comparison between TDC and INDI can be found in [14]. In general, INDI can be viewed as TDC control with a state-dependent gain $\bar{\boldsymbol{G}}(\boldsymbol{x}_0)$ in place of $\bar{\boldsymbol{G}}$. There exist several derivations of INDI in the literature, see for example [3], [10], [14]. INDI is based on a time-scale separation argument, stating that the input rate of change is much faster than the state rate of change. This assumption is absolutely reasonable when INDI is applied as fast inner-loop. The *self-scheduling* property of INDI could enhance system stability. However, for MAVs with fixed geometry and inertial parameters, INDI would have a non-tunable gain, making it less flexible and more dependent on hard-to-obtain parameters such as inertia [3], [4]. Moreover, it could result in very aggressive gains which are not applicable in practice due to, e.g., sensor noise.

For this reason we believe that, if applied to MAVs with constant parameters, TDC is more advisable, as it allows greater tuning flexibility through the gain matrix $\bar{\boldsymbol{G}}$ without requiring the knowledge of mass and inertia. This is similar to the integral gain approach presented in [17], and a relation between TDC and the gains of a PID in incremental form is reported in [14].

C. QP-based TDC

To make the inner-loop controller able to respect constraints, the incremental dynamics (10) is embedded in a QP

$$\min_{\Delta\boldsymbol{u}} . \quad ||\dot{\boldsymbol{x}}_2 - \boldsymbol{\nu}||^2_{2,\boldsymbol{Q}_2} + ||\Delta\boldsymbol{u}||^2_{2,\boldsymbol{U}_2} \tag{14a}$$

$$\text{s.t.} \quad \boldsymbol{u}_{min} \leq \boldsymbol{u}_0 + \Delta\boldsymbol{u} \leq \boldsymbol{u}_{max}, \tag{14b}$$

where $\boldsymbol{Q}_2 \in \mathbb{R}^{6\times6}$ and $\boldsymbol{U}_2 \in \mathbb{R}^{n_p\times n_p}$ are positive definite diagonal matrices, $\dot{\boldsymbol{x}}_2$ is the incremental dynamics (10). In (14), the objective is to track an acceleration $\boldsymbol{\nu}$ defined in (12) while minimizing the *incremental* input and keeping the full input between minimum and maximum limits (the same limit of the NMPC, see (3d)).

By expanding the cost function (14a) and reformulating the constraints, (14) is rewritten as

$$\begin{aligned} \min_{\Delta\boldsymbol{u}} . \quad & \Delta\boldsymbol{u}^\top\boldsymbol{H}\Delta\boldsymbol{u} + \boldsymbol{g}^\top\Delta\boldsymbol{u} \\ \text{s.t.} \quad & \boldsymbol{A}\Delta\boldsymbol{u} \leq \boldsymbol{b} \end{aligned} \tag{15}$$

where

$$\boldsymbol{H} = \tilde{\boldsymbol{G}}^\top\boldsymbol{Q}_2\tilde{\boldsymbol{G}} + \boldsymbol{U}_2, \ \boldsymbol{g} = 2\boldsymbol{Q}_2^\top\tilde{\boldsymbol{G}}^\top(\dot{\boldsymbol{x}}_{2,0} - \boldsymbol{\nu}) \tag{16}$$

$$\boldsymbol{A} = [-\boldsymbol{I} \ \boldsymbol{I}]^\top, \ \boldsymbol{b} = \left[(\boldsymbol{u}_0 - \boldsymbol{u}_{min})^\top \ (-\boldsymbol{u}_0 + \boldsymbol{u}_{max})^\top\right]^\top. \tag{17}$$

Letting $\Delta\boldsymbol{u}^\star$ be the solution of (15), the input to apply to the system at the next step is

$$\boldsymbol{u}^\star = \boldsymbol{u}_0 + \Delta\boldsymbol{u}^\star, \tag{18}$$

which is a vector of n_p squared propeller velocities consistent with the constraint imposed by the NMPC. We call the inner-loop based on (14) TDC-QP.

D. Application to NMPC and Practical Implementation

We explain here how we obtain the reference state and acceleration to be tracked by the inner-loop (computed using (11) or QP (14)). The overall scheme for the TDC-QP inner-loop case is depicted in Fig. 2, where $\boldsymbol{a} \in \mathbb{R}^3$ indicates the

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.

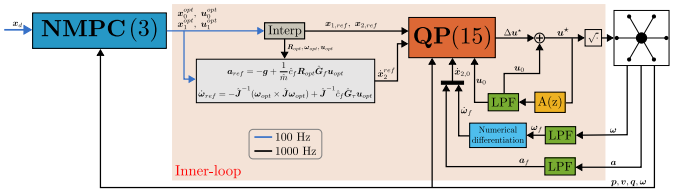


Fig. 2: Control scheme: NMPC outer-loop and TDC-QP inner-loop.

linear acceleration, namely $\dot{\boldsymbol{v}} = \boldsymbol{a}$. If the QP is not used, the corresponding block in Fig. 2 and the sum are replaced by Equation (11). The reference values are obtained from NMPC as follows: at each NLP solution of (3), the algorithm outputs optimal states and inputs for all the shooting nodes along the prediction horizon, namely $\boldsymbol{x}_k^{opt}$ for $k = 0, \dots, N$ and $\boldsymbol{u}_k^{opt}$ for $k = 0, \dots, N-1$, where $\boldsymbol{x}_0^{opt}$ is the same of (3c) and $\boldsymbol{u}_0^{opt}$ is the input that should be applied to the system without inner-loop controller. Since the temporal interval between two consecutive NMPC prediction nodes contains multiple executions of the inner-loop, optimal states and inputs are linearly interpolated between the first and the second shooting nodes, namely between $(\boldsymbol{x}_0^{opt}, \boldsymbol{u}_0^{opt})$ and $(\boldsymbol{x}_1^{opt}, \boldsymbol{u}_1^{opt})$ considering the current time. In Fig. 2, $\boldsymbol{R}_{opt}$ and $\boldsymbol{u}_{opt}$ represent attitude and input computed via interpolation, respectively. Linear and angular reference accelerations $\dot{\boldsymbol{x}}_2^{ref}$ are computed using optimal interpolated states and inputs and inverting the model (2), as shown in Fig. 2.

Note that the *desired* trajectory has no feasibility guarantees and it is filtered by the NMPC to produce feasible (according to nominal model (2)) states and controls, while *reference* trajectory to be tracked by the inner-loop controller is computed through NMPC data and thus inherits the feasibility property.

In practical implementations, signal filtering is crucial since signals from onboard sensors such as the IMU are very noisy. As discussed in [3], [4], [10], all feedback signals must be low-pass filtered with the same filter in order to accumulate the same amount of delay. In Fig. 2, the block LPF indicates a second order Butterworth filter with cutoff frequency f, while low-pass filtered signals are indicated with subscript $\cdot_f$. Moreover, the angular acceleration in (10) (last three elements of $\dot{\boldsymbol{x}}_{2,0}$) cannot be directly measured by sensors and it is thus obtained by numerically differentiating the filtered angular velocity. Since this operation amplifies noise, a good filtering is essential.

Concerning the input feedback $\boldsymbol{u}_0$ in (10), if it cannot be measured (or it cannot be measured fast enough, as in our case), it is necessary to estimate it from previous control commands considering the actuation delay as explained in [10]. This is just a technological limitation which can be solved using different hardware. In this work, we achieve this by applying a first-order filter with a time constant τ_1 to the most recently computed input, which yields the discrete-time transfer function $A(z)$. The estimated input, according to [10], is further filtered with the same filter used for the accelerations in order to synchronize the signals.

Finally, since the computed input $\boldsymbol{u}^\star$ is a vector of n_p squared propeller velocities, we take the elementwise square root to obtain the final command sent to low-level electronics.

V. Experiments

We present here the experimental setup, the details of the implementation on onboard hardware and the final results. The purpose of the experiments is to show that the proposed TDC-QP control scheme offers the same advantages of other inner-loop controllers for NMPC *while* guaranteeing stability in case of saturation and model mismatches, the latter condition being unavoidable in real robots. The videos related to the proposed experiments can be found in the multimedia material.

A. Experimental Setup

The proposed control algorithm is tested on the fully-actuated hexarotor with fixedly tilted propellers shown in Fig. 1 whose design has been introduced in [18]. The platform weights $3.05\,\mathrm{kg}$ including batteries and $n_p = 6$. Note that this vehicle is fully-actuated and not omnidirectional, and the forces that it can exert laterally (in the $\boldsymbol{x}_R$-$\boldsymbol{y}_R$ plane) are limited and smaller than the one along $\boldsymbol{z}_R$, as clearly visible in the experimental part of [19], where the same robot is used. The robot is equipped with a NUC13 onboard computer running Ubuntu. The flight controller runs custom firmware and is equipped with IMU (accelerometer and gyroscope). The experiments are performed in an indoor arena equipped with motion capture system providing position and attitude data at $200\,\mathrm{Hz}$ which are fused with IMU data sampled at $1\,\mathrm{kHz}$ by the state estimator. As previously mentioned, the final command sent to low-level electronics are desired rotors speeds, which are tracked in closed-loop by a control algorithm [20] embedded in the Electronic Speed Controllers (ESCs).

The robot is controlled to track a trajectory which may be unfeasible (i.e., lead to input saturation) due to excessive accelerations or to heavy model mismatches. Note that minimal model mismatches are always present, even in nominal conditions, because of air turbulences and imprecise assembly.

B. Implementation Details

The NMPC is implemented with acados [21] Python interface using the Real Time Iteration (RTI) scheme [22], and runs on the high-level onboard computer at $100\,\mathrm{Hz}$ with a $1\,\mathrm{s}$ prediction horizon and 20 shooting nodes. Nominal dynamics (2) is discretized using a fourth order Runge-Kutta integrator. State estimation, inner-loop controller (TDC and TDC-QP), and interfaces with a motion tracking system (mocap) and low-level electronics run onboard at $1\,\mathrm{kHz}$ within the open-source framework *Telekyb3*[1] . The control algorithm described in this paper is implemented in a novel component called `tdopt`, where PROXQP [23] is used as QP solver. In order to compare the performance of different controllers, it is possible to use TDC inner-loop control without QP computing the input with (11) or to not use the inner-loop at all. The first option is comparable to INDI as discussed in Sec. IV-B, while in the latter case the control input computed by (3) is directly sent to low-level electronics.

[1]https://git.openrobots.org/projects/telekyb3

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.

Control	Param.	Value	Control	Param.	Value
NMPC	$\boldsymbol{Q_p}$	$250 \cdot \boldsymbol{I}_3$	TDC-QP	$\bar{\boldsymbol{g}}_{f_{xy}}$	$0.003 \cdot \boldsymbol{I}_2$
NMPC	$\boldsymbol{Q_q}$	$10 \cdot \boldsymbol{I}_3$	TDC-QP	$\bar{g}_{f_z}$	0.0006
NMPC	$\boldsymbol{Q_v}$	$5 \cdot \boldsymbol{I}_3$	TDC-QP	$\bar{\boldsymbol{g}}_\tau$	$0.03 \cdot \boldsymbol{I}_3$
NMPC	$\boldsymbol{Q_\omega}$	$0.1 \cdot \boldsymbol{I}_3$	TDC-QP	$\boldsymbol{K_p}$	$9 \cdot \boldsymbol{I}_3$
NMPC	$\boldsymbol{U}$	$10^{-8} \cdot \boldsymbol{I}_6$	TDC-QP	$\boldsymbol{K_q}$	$9 \cdot \boldsymbol{I}_3$
NMPC	N	20	TDC-QP	$\boldsymbol{K_v}$	$4.242 \cdot \boldsymbol{I}_3$
NMPC	dt	50 ms	TDC-QP	$\boldsymbol{K_\omega}$	$4.242 \cdot \boldsymbol{I}_3$
TDC-QP	$\boldsymbol{Q}_2$	$\boldsymbol{I}_6$	TDC-QP	f	5 Hz
TDC-QP	$\boldsymbol{U}_2$	$10^{-8} \cdot \boldsymbol{I}_6$	TDC-QP	τ_1	0.02

TABLE I: Control gains and parameters.

The control gains and parameters used for the experiments are reported in Table I, where the gain matrix has been divided in $\bar{\boldsymbol{G}} = \text{diag}(\bar{\boldsymbol{g}}_{f_{xy}}, \bar{g}_{f_z}, \bar{\boldsymbol{g}}_\tau)$ in order to distinguish gain in horizontal force, vertical force and torque, respectively. The desired input $\boldsymbol{u}_{k,d}$ in cost function (3a) is set to the hovering value.

C. Experimental Results

We performed a set of experiments with different conditions in order to validate the proposed TDC-QP inner-loop and compare it to other controllers. For each scenario (numbered from 1 to 4), three controls are tested and compared: NMPC without inner-loop control; NMPC + TDC inner-loop without QP computed with (11); NMPC + TDC-QP inner-loop computed with (15), with results in Table II. After a gain tuning process, to ensure comparable performance to TDC-QP, the second controller is implemented with $\bar{\boldsymbol{g}}_\tau = 0.015\ \boldsymbol{I}_3$ while the other gains are as in Table I. Its limitations are still highlighted by experiments with input saturation.

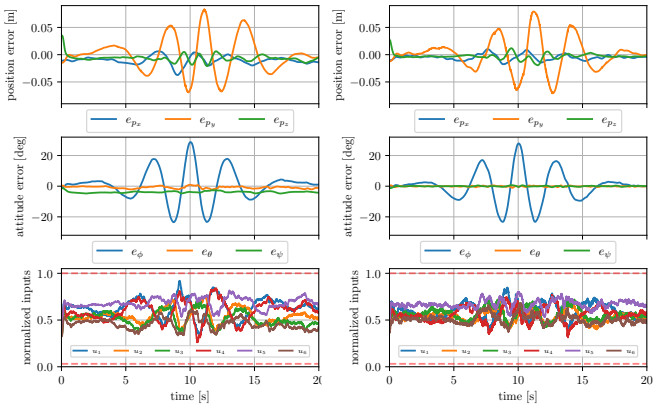
The trajectory chosen for the experiments is a double symmetric chirp along $\boldsymbol{y}_W$. This seems an appropriate choice since it can heavily affect also the attitude of the MAV and it allows to trigger input saturation even in a small space by properly changing its parameters. More in detail, the desired position is $\boldsymbol{p}_d(t) = [0\ p_y(t)\ \bar{z}]^\top$, where $\bar{z}$ is the desired takeoff height and

$$p_y(t) = A\sin(\xi(t)t),\ \xi(t) = \begin{cases} \xi t & \text{if } t \in [0, \frac{\bar{t}}{2}[\\ \xi(\bar{t} - t) & \text{if } t \in [\frac{\bar{t}}{2}, \bar{t}] \end{cases} \quad (19)$$

where A is the amplitude and ξ is the slope used to transition from the initial frequency $\xi(0) = 0\,\text{Hz}$ to $\xi(\frac{\bar{t}}{2}) = f_m$ to finally $\xi(\bar{t}) = 0\,\text{Hz}$.

The desired velocity $\boldsymbol{v}_d(t)$ is defined accordingly with the position, while desired attitude and angular velocities are set to zero. However, NMPC cost function weights give more importance to position tracking than attitude tracking, so the hexarotor tilts when the required lateral force exceeds the platform bounds.

By varying chirp parameters and experiments conditions, we consider four scenarios with results reported in Table II and discussed in the following. In Table II, columns "M" and "S" specify respectively if model mismatches have been added to the hexarotor beside the minimal ones and if it reaches heavy input saturation. The chirp amplitude is $A = 1\,\text{m}$ for each scenario. The controllers performance are evaluated computing the RMS values of position tracking error $\boldsymbol{e}_p$ and attitude tracking error converted as roll, pitch and yaw errors $\boldsymbol{e}_q = [e_\phi\ e_\theta\ e_\psi]^\top$ for better clarity. The input is depicted normalized dividing by its maximum achievable value. The red dashed lines in the input plots represents minimum and maximum values. A red background in input plots indicates input saturation. Motors are numbered from 1 to 6 counter-clockwise, with motor 1 on the positive $\boldsymbol{x}_R$ axis. Takeoff and landing are not depicted in the plots and not considered for errors computation.



(a) NMPC (b) NMPC+TDC-QP

Fig. 3: Scenario 1 with no saturation nor added mismatches.

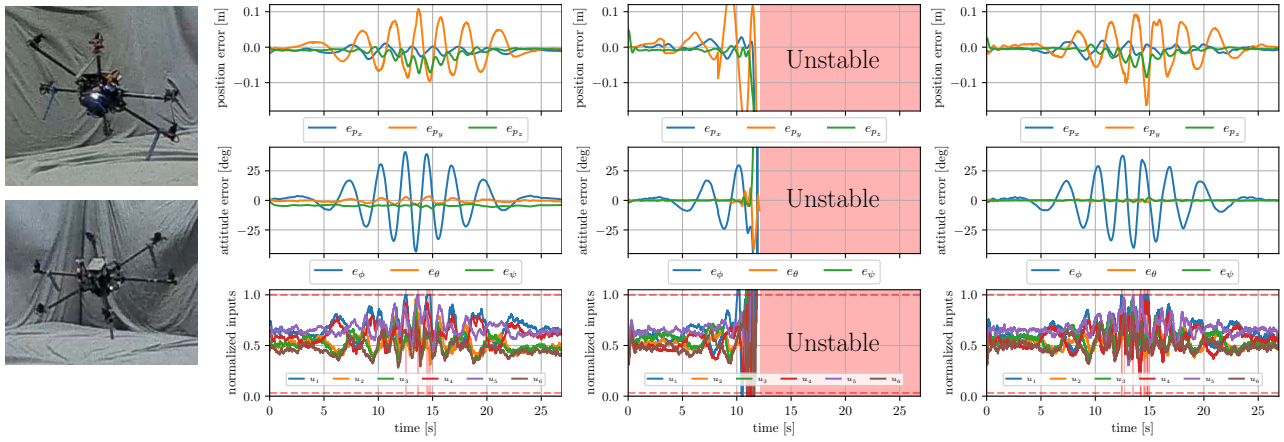
1) Scenario 1 - nominal conditions: we first test the algorithms without additional model mismatches. Only minimal (unavoidable) mismatches are present and there is no input saturation. The three controllers perform similarly and they all preserve system stability. From Table II and Fig. 3 we can see that, even if there are turbulences due to air bouncing against the walls, the NMPC is able to reject the unknown disturbance also without inner-loop controller. However, there is a constant error in yaw which is compensated with both inner-loop controllers. The reason for this yaw offset may be the imperfect assembly of the hexarotor, which does not lead to a perfect compensation of the internal forces. In general, from the RMS errors, it is clear that an inner-loop controller decreases the attitude error, as the inertia is one of the most uncertain parameters in MAVs, while the mass is easier to measure. The fact that constant errors are reduced to zero suggests that acceleration-based inner-loop controllers (both with and without QP) embed an integral action, as discussed in [14], [17]. Lastly, another advantage of the TDC-QP inner-loop is that it leads to more uniform inputs as shown in Fig. 3(b), because it penalizes high input increments.

2) Scenario 2 - aggressive trajectory: we test the controllers with a trajectory which leads to input saturation without additional model mismatches beside the minimal ones. As shown in Table II and Fig. 4, NMPC and NMPC+TDC-QP performs similarly, with the advantage of the latter of reducing attitude and vertical position errors as in previous experiments. This control scheme is again stable despite the fact it reaches saturation multiple times, as depicted in Fig. 4(d) and Fig. 4(b). The TDC inner-loop without QP becomes unstable in the most aggressive part of the trajectory and the experiment

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.

Scenario	M	S	f_m [Hz]	$\bar{t}$ [s]	Tracking error	NMPC	NMPC+TDC no QP	NMPC+TDC-QP
1	✗	✗	0.40	20.0	$\boldsymbol{e}_{p,RMS}$ [m]	$[0.013, 0.032, 0.009]^\top$	$[0.014, \mathbf{0.028}, 0.010]^\top$	$[\mathbf{0.007}, 0.030, \mathbf{0.006}]^\top$
					$\boldsymbol{e}_{q,RMS}$ [deg]	$[10.133, 1.047, 3.876]^\top$	$[10.244, 0.618, 0.372]^\top$	$[\mathbf{9.820}, \mathbf{0.283}, \mathbf{0.261}]^\top$
2	✗	✓	0.52	26.9	$\boldsymbol{e}_{p,RMS}$ [m]	$[0.013, \mathbf{0.040}, 0.021]^\top$	CRASH	$[\mathbf{0.012}, 0.041, \mathbf{0.018}]^\top$
					$\boldsymbol{e}_{q,RMS}$ [deg]	$[15.430, 1.350, 4.443]^\top$	CRASH	$[\mathbf{14.956}, \mathbf{0.458}, \mathbf{0.438}]^\top$
3	✓	✗	0.40	20.0	$\boldsymbol{e}_{p,RMS}$ [m]	$[\mathbf{0.012}, 0.032, 0.026]^\top$	$[0.013, 0.029, 0.011]^\top$	$[0.013, \mathbf{0.029}, \mathbf{0.011}]^\top$
					$\boldsymbol{e}_{q,RMS}$ [deg]	$[10.673, 1.631, 5.230]^\top$	$[10.443, \mathbf{0.552}, \mathbf{0.384}]^\top$	$[\mathbf{10.104}, 0.558, 0.385]^\top$
4	✓	✓	0.40	20.0	$\boldsymbol{e}_{p,RMS}$ [m]	$[\mathbf{0.007}, 0.033, 0.030]^\top$	CRASH	$[0.012, \mathbf{0.032}, \mathbf{0.008}]^\top$
					$\boldsymbol{e}_{q,RMS}$ [deg]	$[10.734, 2.107, 5.200]^\top$	CRASH	$[\mathbf{10.387}, \mathbf{0.989}, \mathbf{0.871}]^\top$

TABLE II: Experiments results for the tested scenarios and controllers.



(a) Snapshots (b) NMPC (c) NMPC+TDC without QP (d) NMPC+TDC-QP

Fig. 4: Scenario 2 with aggressive trajectory, no added mismatches (only minimal ones).

is manually interrupted (red background in Fig. 4(c)). In an ideal case (achievable only in simulation) this situation would not lead to a crash, because the commands from NMPC would be tracked without exceeding input limitations as real and nominal models coincide. This experiment highlights the importance of a *safe* inner-loop such as TDC-QP even with minimal, unavoidable mismatches. To further investigate and validate this, we tested the same controller in a simulation where nominal and real models coincides and the inner-loop controller without QP had comparable performance to TDC-QP, maintaining stability.

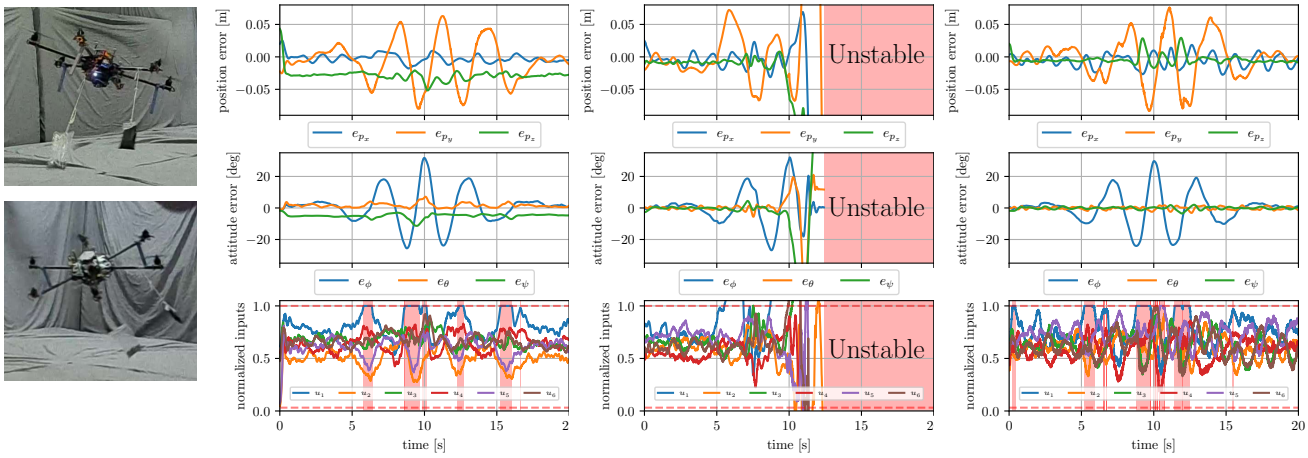
3) Scenario 3 - additional model mismatches: to add model mismatches to the minimal ones, a 350 g mass is attached with a rope near the CoM of the robot and acts as a pendulum. The chirp trajectory is the same of scenario 1. From Table II one can notice that, since the NMPC is not aware of the added mass, it has a constant offset in the vertical position and the yaw error is increased wrt experiment 1. The results show that the TDC-based inner-loop controllers (with and without QP) are able to greatly reduce these offsets. In particular, the two inner-loops perform very similarly on both position and attitude trackings. These results are similar to Scenario 1 and for space reasons we do not show the plots here.

4) Scenario 4 - heavy model mismatches and saturation: we test here the advantage of TDC-QP inner-loop control in case of model mismatch and trajectory leading to heavy input saturation. The benefits of inner-loop tracking are already known, but we show here how TDC-QP is able to provide lower tracking errors *while* preserving system stability also in this challenging conditions, which is the main focus of this paper. These conditions may represent a system of which a very coarse model is available. The desired chirp trajectory is the same of scenario 1 and 3. To increase the model mismatch beyond the minimal one, some masses are attached to the hexarotor with ropes in a pendulum-like style. In particular, there are a 300 g mass attached slightly below the CoM, and a 220 g mass attached along the $\boldsymbol{x}_R$ axis to the arm connecting motor 1 to the frame, as shown in Fig. 5(a). Note that the masses act as a pendulum and they introduce not only unknown time-varying forces, but also torques. From Table II and Fig. 5, it is evident that the TDC-QP inner-loop is able to greatly reduce the tracking errors, in particular in vertical position and yaw, while handling the saturation of the input. TDC without QP, instead, becomes unstable and leads to a crash (red background in Fig. 5(c)). These behaviors are also shown in the attached video.

VI. Conclusion and Future Work

Tracking aggressive trajectories under strong model uncertainties still represents a challenge in Aerial Robotics, as common inner-loop controllers used to improve accuracy can cause input saturation and, thus, instability. To address this, we propose a TDC-based fast inner-loop for NMPC trajectory tracking. Our approach retains the benefits of acceleration-based inner-loop controllers *while* ensuring actuation constraints align with the outer NMPC loop, preserving stability under saturation and model mismatches. This is achieved by

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.



(a) Snapshots (b) NMPC (c) NMPC+TDC without QP (d) NMPC+TDC-QP

Fig. 5: Scenario 4 with heavy added mismatches and saturation.

formulating dynamics in incremental form and embedding it in a QP, considering filters delays and interpolating NMPC data. Experiments on a fully-actuated hexarotor demonstrate superior safety performance in challenging conditions, handling input saturation where others fail.

In the future, we plan to apply TDC-QP inner-loop to platforms such as torque-controlled manipulators and aerial manipulators, where we could achieve *model-free* control [16]. Since in those systems the inertia varies with the state, a comparison with INDI inner-loop could be performed. As hardware improvements, to enhance inner-loop reliability, we will further investigate vibration damping on IMU and use more performant ESCs capable of providing fast rotors speed feedback. This will eliminate the need to estimate the actuation delays in the input feedback.

REFERENCES

[1] A. Ollero, M. Tognon, A. Suarez, D. Lee, and A. Franchi, "Past, present, and future of aerial robotic manipulators," *IEEE Transactions on Robotics*, vol. 38, no. 1, pp. 626–645, 2021.

[2] T. Lee, M. Leok, and N. H. McClamroch, "Geometric tracking control of a quadrotor uav on se (3)," in *49th IEEE conference on decision and control (CDC)*. IEEE, 2010, pp. 5420–5425.

[3] E. Tal and S. Karaman, "Accurate tracking of aggressive quadrotor trajectories using incremental nonlinear dynamic inversion and differential flatness," *IEEE Transactions on Control Systems Technology*, vol. 29, no. 3, pp. 1203–1218, 2020.

[4] S. Sun, A. Romero, P. Foehn, E. Kaufmann, and D. Scaramuzza, "A comparative study of nonlinear mpc and differential-flatness-based control for quadrotor agile flight," *IEEE Transactions on Robotics*, vol. 38, no. 6, pp. 3357–3373, 2022.

[5] H. Nguyen, M. Kamel, K. Alexis, and R. Siegwart, "Model predictive control for micro aerial vehicles: A survey," in *2021 European Control Conference (ECC)*. IEEE, 2021, pp. 1556–1563.

[6] D. Bicego, J. Mazzetto, R. Carli, M. Farina, and A. Franchi, "Nonlinear model predictive control with enhanced actuator model for multi-rotor aerial vehicles with generic designs," *Journal of Intelligent & Robotic Systems*, vol. 100, no. 3, pp. 1213–1247, 2020.

[7] L. Peric, M. Brunner, K. Bodie, M. Tognon, and R. Siegwart, "Direct force and pose nmpc with multiple interaction modes for aerial push-and-slide operations," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021, pp. 131–137.

[8] K. D. Young, V. I. Utkin, and U. Ozguner, "A control engineer's guide to sliding mode control," *IEEE transactions on control systems technology*, vol. 7, no. 3, pp. 328–342, 1999.

[9] T. Başar and P. Bernhard, *H-infinity optimal control and related minimax design problems: a dynamic game approach*. Springer Science & Business Media, 2008.

[10] E. J. Smeur, Q. Chu, and G. C. De Croon, "Adaptive incremental nonlinear dynamic inversion for attitude control of micro air vehicles," *Journal of Guidance, Control, and Dynamics*, vol. 39, no. 3, pp. 450–461, 2016.

[11] X. Wang, E.-J. Van Kampen, Q. Chu, and P. Lu, "Stability analysis for incremental nonlinear dynamic inversion control," *Journal of Guidance, Control, and Dynamics*, vol. 42, no. 5, pp. 1116–1129, 2019.

[12] D. Hanover, P. Foehn, S. Sun, E. Kaufmann, and D. Scaramuzza, "Performance, precision, and payloads: Adaptive nonlinear mpc for quadrotors," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 690–697, 2021.

[13] K. Youcef-Toumi and S.-T. Wu, "Input/output linearization using time delay control," 1992.

[14] B. P. Acquatella and Q. P. Chu, "Agile spacecraft attitude control: An incremental nonlinear dynamic inversion approach," *IFAC-PapersOnLine*, vol. 53, no. 2, pp. 5709–5716, 2020.

[15] Y. Wang, M. Leibold, J. Lee, W. Ye, J. Xie, and M. Buss, "Incremental model predictive control exploiting time-delay estimation for a robot manipulator," *IEEE Transactions on Control Systems Technology*, vol. 30, no. 6, pp. 2285–2300, 2022.

[16] Y. Wang, Y. Liu, M. Leibold, M. Buss, and J. Lee, "Hierarchical incremental mpc for redundant robots: a robust and singularity-free approach," *IEEE Transactions on Robotics*, 2024.

[17] M. Hamandi, M. Tognon, and A. Franchi, "Direct acceleration feedback control of quadrotor aerial vehicles," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 5335–5341.

[18] S. Rajappa, M. Ryll, H. H. Bülthoff, and A. Franchi, "Modeling, control and design optimization for a fully-actuated hexarotor aerial vehicle with tilted propellers," in *2015 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2015, pp. 4006–4013.

[19] A. Franchi, R. Carli, D. Bicego, and M. Ryll, "Full-pose tracking control for aerial robotic systems with laterally bounded input force," *IEEE Transactions on Robotics*, vol. 34, no. 2, pp. 534–541, 2018.

[20] A. Franchi and A. Mallet, "Adaptive closed-loop speed control of bldc motors with applications to multi-rotor aerial vehicles," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, 2017, pp. 5203–5208.

[21] R. Verschueren, G. Frison, D. Kouzoupis, J. Frey, N. van Duijkeren, A. Zanelli, B. Novoselnik, T. Albin, R. Quirynen, and M. Diehl, "acados – a modular open-source framework for fast embedded optimal control," *Mathematical Programming Computation*, 2021.

[22] S. Gros, M. Zanon, R. Quirynen, A. Bemporad, and M. Diehl, "From linear to nonlinear mpc: bridging the gap via the real-time iteration," *International Journal of Control*, vol. 93, no. 1, pp. 62–80, 2020.

[23] A. Bambade, S. El-Kazdadi, A. Taylor, and J. Carpentier, "Prox-qp: Yet another quadratic programming solver for robotics and beyond," in *RSS 2022-Robotics: Science and Systems*, 2022.