Controller and Trajectory Optimization for a Quadrotor UAV with Parametric Uncertainty
Ali Srour, Antonio Franchi, Paolo Robuffo Giordano

To cite this version:

HAL Id: hal-04216628
https://hal.science/hal-04216628
Submitted on 25 Sep 2023

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L’archive ouverte pluridisciplinaire HAL, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d’enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Distributed under a Creative Commons Attribution 4.0 International License
Controller and Trajectory Optimization for a Quadrotor UAV with Parametric Uncertainty

Ali Srour¹, Antonio Franchi²,³,⁴, Paolo Robuffo Giordano¹

I. INTRODUCTION

The requirement for autonomous/intelligent systems such as robots to operate in unpredictable and/or uncertain real-world conditions is one of the biggest challenges in our community. In fact, whether directly or indirectly, robot decisions and control actions are based on a model of the "world", which is oftentimes only a rough representation of the physical reality. When considering motion tasks, one of the common sources of uncertainty is in the robot model parameters that can be hard to measure or also vary at runtime. Adaptive [1] or robust control [2] techniques are typical ways to deal with parametric uncertainty by either trying to estimate the parameters online, or by trading off performance for robustness vs. parametric uncertainty. Also, Model Predictive Control [3], [4] predicts system behavior and optimizes a cost function over a finite time horizon using a dynamic model of the system and its environment. However, real-time rescheduling for optimization at each time step can result in significant computational overhead. Another recent approach to deal with these issues has been proposed in [5]–[10] where several metrics based on the notion of closed-loop state/input sensitivity have been introduced. These metrics are able to quantify how uncertainties in the parameters of a robot model can affect the evolution of the robot states and inputs in closed-loop and as a function of the reference trajectory being tracked (i.e., the 'motion task'). One can then attempt to increase robustness against uncertain parameters by planning a suitable feedforward (or desired) trajectory that minimizes the sensitivity metrics, thereby generating an intrinsically-robust and control-aware motion plan. These ideas have also been extended to the problem of combining state observability with parametric robustness in [11], and in [12] a recent extension is proposed for obtaining the 'tubes' of perturbed trajectories given a known range of parametric deviation.

All these previous works have shown very promising results by focusing on the optimization of the reference trajectory to be tracked by the robot under suitable state and input constraints. However, one may also consider the concurrent optimization of the gains of the chosen control strategy (besides the reference trajectory) for the purpose of minimizing a sensitivity metric. In this paper we then consider this latter possibility applied, as case study, to a tracking task for a quadrotor UAV under three popular control strategies: a near-hovering (or linear) controller, a nonlinear geometric controller, and a sliding-mode controller. We present and discuss the results of a thorough statistical analysis aimed at comparing the three controllers in different conditions (non-optimized, by only optimizing for the reference trajectory, by optimizing for the control gains only and by optimizing for both the trajectory and the control gains). The aim is twofold: (i) to show how gain optimization (or tuning) can be done while taking into account a concrete robustness metric (the state sensitivity vs. parametric uncertainties) and (ii) to get statistical insights
into the behavior, performance and robustness of the three controllers in different conditions.

The rest of the paper is structured as follows: in Sect. II we recall the main notions related to the closed-loop state sensitivity. Then in Sect. III we present the optimization problems considered in this work, and we detail the quadrotor model and the three tracking controller. Section IV discusses the results of the statistical analysis on the controller performance, and Sect. V concludes the paper and draws some future perspectives.

II. PRELIMINARIES

In this section we briefly summarize for the reader’s convenience the main notions related to the closed-loop state sensitivity introduced in [5], [6] and recently extended in [12]. We consider a robot model
\[ \dot{x} = f(x, u, p) \]  
where \( x \in \mathbb{R}^{n_x} \) is the state, \( u \in \mathbb{R}^{n_u} \) the control inputs and \( p \in \mathbb{R}^{n_p} \) a vector of (possibly uncertain) model parameters. We assume some quantity of interest \( s(x) \in \mathbb{R}^{n_s} \) (e.g., the position of the robot center) needs to track a desired trajectory \( s_d(t, a) \) defined for \( t \in [t_0, t_f] \) and function of a finite set of parameters \( a \in \mathbb{R}^{n_a} \). For this task a tracking controller is used
\[ \begin{align*}
\dot{\xi} &= g(\xi, x, a, p_c, k_c, t) \\
\dot{u} &= h(\xi, x, a, p_c, k_c, t)
\end{align*} \]
which is evaluated at a nominal value \( p_c \) for the uncertain parameters \( p \). The vector \( \xi \in \mathbb{R}^{n_\xi} \) represents the possible internal controller states (e.g., an integral action) and \( k_c \in \mathbb{R}^{n_k} \) is the vector of control gains.

The state sensitivity for the closed-loop system (1–2) is defined as
\[ \Pi(t) = \left. \frac{\partial x(t)}{\partial p} \right|_{p=p_c} \]
and the input sensitivity
\[ \Theta(t) = \left. \frac{\partial u(t)}{\partial p} \right|_{p=p_c} \]  

Matrix \( \Pi(t) \) quantifies how variations in the parameters \( p \) around a nominal value \( p_c \) will affect the evolution of the states \( x \) (in closed-loop). Analogously, matrix \( \Theta(t) \) relates variations in \( p \) to variations in the inputs \( u \). A closed-form expression for matrices \( \Pi(t) \) and \( \Theta(t) \) is in general not available, but in [5], [6] it is shown how these two quantities can be easily evaluated via forward integration of a differential equation along the system trajectories.

Matrices \( \Pi(t) \) and \( \Theta(t) \) can be used for several purposes such as optimization of the reference trajectory \( s_d(t, a) \) for producing minimally sensitive motion plans (by minimizing some combined norm of \( \Pi(t) \) and \( \Theta(t) \) as done in [5], [6]). An extension of these ideas has been recently proposed in [12] where a suitable weighted norm for the state sensitivity \( \Pi(t) \) is introduced. Assume that each parameter \( p_i \) can vary in a given range \( \delta p_i \) centered at a nominal \( p_c \),
\[ p_i \in [p_c - \delta p_i, p_c + \delta p_i] \]
and define the diagonal weight matrix \( W = \text{diag}(\delta p_i^2) \). Letting \( \Delta p = p - p_c \), an ellipsoid in parameter space centered at \( p_c \) and with semi-axes \( \delta p_i \) has equation
\[ \Delta p^T W^{-1} \Delta p = 1. \]  

From (6) and (3–4) one can obtain the corresponding ellipsoids in state space
\[ \Delta x^T (\Pi W \Pi^T)^{-1} \Delta x = 1 \]
and in input space
\[ \Delta u^T (\Theta W \Theta^T)^{-1} \Delta u = 1, \]  
which represents the largest (worst-case) deviation of the states \( x \) assuming a parametric uncertainty as in (5). Furthermore, one can also exploit (7–8) for obtaining the tubes of perturbed trajectories for the individual components of the states and the inputs. By again referring to [12] for all details, for each direction of interest in the state space one can obtain the ‘tube radius’ \( r_i(t) \) such that
\[ q_{nom,i}(t) - r_i(t) \leq q_i(t) \leq q_{nom,i}(t) + r_i(t). \]

III. OPTIMIZATION PROBLEM

As explained in Sect. I, in this work we are interested in comparing the performance of three typical trajectory tracking controllers for a quadrotor UAV in the context of minimally sensitive trajectory optimization. To this end, and considering the various quantities introduced in Sect. II, we consider a first trajectory optimization problem
\[ a^* = \arg \min_a || \Pi(t_f)||_W \quad \text{s.t.} \quad M a = b \]
\[ U_{\min,i} \leq u_{nom,i}(t) - r_i(t) \quad \forall i \forall \forall t \in [t_0, t_f] \]
\[ u_{nom,i}(t) + r_i(t) \leq U_{\max,i} \quad \forall i \forall \forall t \in [t_0, t_f]. \]

We seek to find the optimal value \( a^* \) for the shape parameter \( a \) of the reference trajectory \( s_d(a, t) \) for minimizing the weighted norm (9) at the final time \( t_f \). The constraints consist of given initial/final conditions for \( s_d(a, t) \), represented by the linear constraints \( M a = b \), and constraints that bound the envelope of perturbed inputs within actuation limits
$U_{\text{min},i} \leq U_{\text{max},i}$, ensuring that the tracking of the optimized reference trajectory will be feasible for any value of the uncertain parameters $p$ in the range (5). These constraints leverage the ‘tubes’ as in (10) but are evaluated for the inputs. Variants of this problem have already been considered in [5], [6], [12] but by always focusing on a single control strategy.

We also consider a second optimization problem

$$k^*_c = \arg \min_{k_c} \| \Pi(t_f) \|_W$$

$$\text{s.t. } Ma = b$$

$$K_{\text{min},i} \leq k_c \leq K_{\text{max},i}$$

$$U_{\text{min},i} \leq u_{\text{nom},i}(t) - r_i(t) \forall i \forall t \in [t_0, t_f]$$

$$u_{\text{nom},i}(t) + r_i(t) \leq U_{\text{max},i} \forall i \forall t \in [t_0, t_f]$$

where we optimize w.r.t. the control gains $k_c$ inside suitable bounds $0 < K_{\text{min},i} \leq K_{\text{max},i}$. We then consider a third optimization problem that optimizes both the shape parameter $\alpha$ and the control gains $k_c$, subject to the same constraints as in equation (12).

$$(\alpha^*, k^*_c) = \arg \min_{\alpha, k_c} \| \Pi(t_f) \|_W$$

$$\text{s.t. } Ma = b$$

$$K_{\text{min},i} \leq k_c \leq K_{\text{max},i}$$

$$U_{\text{min},i} \leq u_{\text{nom},i}(t) - r_i(t) \forall i \forall t \in [t_0, t_f]$$

$$u_{\text{nom},i}(t) + r_i(t) \leq U_{\text{max},i} \forall i \forall t \in [t_0, t_f]$$

We note that in the previous problems we focus on the minimization of the state sensitivity norm at the final time $t_f$. In other words, we look for the combination of reference trajectory and/or control gains that will result in the largest tracking accuracy at the final time (for e.g., reaching at best a target location). Of course, different choices are also possible such as minimization of (the integral of) the sensitivity norm along the whole trajectory (for increasing the average tracking accuracy during motion). Note also that additional constraints, such as obstacle avoidance, could be easily added to the optimization problem by exploiting the tubes on the states, see [13] for an example in this sense. One can also optimize the sensitivity at specific waypoints along the trajectory where, for instance, the effects of uncertainties in high speed regimes can be minimized.

A. Quadrotor Model

Let $x = (r, v, q, \omega) \in \mathbb{R}^6 \times \mathbb{S}^3 \times \mathbb{R}^3$ be the quadrotor state vector consisting of the world-frame position $r = (r_x, r_y, r_z)$ and velocity $v = (v_x, v_y, v_z)$, the rotation from body to world frame represented by a unit quaternion $q = (q_w, q_x, q_y, q_z)$ and the body-frame angular velocity $\omega = (\omega_x, \omega_y, \omega_z)$. The quadrotor model is

$$\dot{r} = v$$

$$\dot{v} = \frac{f}{m} \left[ \frac{2(q_yq_x + q_zq_w)}{2q_yq_x - q_zq_w} \right] - g$$

$$\dot{q} = \frac{1}{2} \begin{bmatrix} 0 \\ w \end{bmatrix} \otimes q$$

$$\dot{w} = \Gamma^{-1}(\tau - w \times Iw)$$

In this model, $f$ and $\tau$ denote the total thrust and body-frame moment, $g = (0, 0, g)$ is the gravity vector in world frame, and $m$ and $I$ represent the mass and body-frame inertia. Although quaternions are used to represent the quadrotor orientation in model (14), the conversion of quaternions to the roll $\phi$, pitch $\theta$, and yaw $\psi$ angles is also considered in two of the control strategies.

Let $w_i$ be the squared velocity of the $i$-th propeller and define the quadrotor control input $u = (w_1, \ldots, w_d)$. An allocation matrix is used to relate the inputs $u$ (i.e., the squared propeller speeds) to the thrust/torques $(f, \tau)$.

$$f = \begin{bmatrix} \frac{1}{k_t} & -g_x & -g_y & -g_z \\ -g_x & \frac{1}{k_t} & -g_y & -g_z \\ -g_y & -g_x & \frac{1}{k_t} & -g_z \\ -g_z & -g_x & -g_y & \frac{1}{k_t} \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \\ w_3 \\ w_4 \end{bmatrix}$$

$$\tau = T u$$

The aerodynamic coefficients of the propellers are denoted by $k_f$ and $k_t$, the $x$ and $y$ coordinates of the barycenter in the body frame are denoted by $g_x$ and $g_y$, and the length of the propeller arms is denoted by $L$ (see [14] for a more detailed derivation). These coefficients are difficult to measure reliably and may change at runtime depending on the flight regime. Therefore, the vector of uncertain parameters considered in this work is $p = (k_f, k_t, g_x, g_y)$.

B. Near-Hovering Controller (NH)

The first control strategy considered is the near-hovering or linear controller, commonly used for its ease of implementation and tuning. However, it only performs well and maintains stability near a hovering state with small roll/pitch angles [15]. Tracking is done using a 4D reference trajectory $s_d(t) = (r_d(t), \psi_d(t))$, which includes the quadrotor 3D position $r_d(t) = (x_d(t), y_d(t), z_d(t))$ and yaw angle $\psi_d(t)$. The final control equations are

$$\ddot{x} = \dot{r}_d + k_{s_d}(\dot{r}_d - \dot{x}) + k_{p_s}(r_d - x) + k_{i_s} \int (r_d - x) dt$$

$$\ddot{y} = \dot{r}_d + k_{s_d}(\dot{r}_d - \dot{y}) + k_{p_y}(r_d - y) + k_{i_y} \int (r_d - y) dt$$

$$\ddot{\phi} = -\arcsin \left( \frac{m}{f} \left( -g_x x + \cos(\phi) \dot{y} \right) \right)$$

$$\ddot{\theta} = \arcsin \left( \frac{m}{f \cos(\phi)} \left( g_y \dot{x} - \sin(\phi) \dot{y} \right) \right)$$

$$\dot{f} = \frac{m}{\cos(\phi) \cos(\theta)} \left( g + \dot{r}_d + k_{s_d}(\dot{r}_d - \dot{z}) + k_{p_z}(r_d - z) + k_{i_z} \int (r_d - z) dt \right)$$

$$\tau_x = -k_{s_d} \omega_x + k_{p_x}(\dot{\phi} - \phi)$$

$$\tau_y = -k_{s_d} \omega_y + k_{p_y}(\dot{\theta} - \theta)$$

$$\tau_z = -k_{s_d} \omega_z + k_{p_z}(\dot{\psi} - \psi)$$

Finally, the quadrotor inputs are computed as

$$u = T^{-1} \begin{bmatrix} f \\ \tau \end{bmatrix}$$
where the the allocation matrix $T$ from (15) is evaluated at the nominal parameters $p_c$.

The vector of control gains $k_c$ considered in problem (12–13) for this controller consists of all the 15 gains in (16), that is, the three gains $(k_{b_1}, k_{b_2}, k_{b_3})$ for the $x, y$ and $z$ channels, and the two gains $(k_{s_1}, k_{s_2})$ for the $\phi, \theta$ and $\psi$ channels.

C. 3D Geometric Controller (LEE)

The 3D Geometric controller [16] is a popular control strategy that differs from the near-hovering controller by not relying on approximations of the quadrotor dynamics and providing almost global stability. However, it is more challenging to tune than the near-hovering controller.

Let $\mathbf{R}(q) \in SO(3)$ be the rotation matrix associated to the quaternion $q$, and let $e_3 = (0, 0, 1)$. The idea behind this control strategy is to design a desired thrust vector $f \mathbf{R}_d e_3$ for stabilizing the translational dynamics (i.e., tracking the reference $r_d(t)$) while the torques $\boldsymbol{\tau}$ try to align the quadrotor $z$ axis $\mathbf{R}_e e_3$ with the desired one and also track the reference signal for the yaw angle $\psi_d(t)$. The final controller equations are (see [16])

$$
\begin{align*}
\mathbf{e}_r &= r_d - r, \quad \mathbf{e}_v = \dot{r}_d - v \\
\mathbf{b}_{3d} &= \begin{pmatrix} K_r \mathbf{e}_r + K_v \mathbf{e}_v + K_i \int \mathbf{e}_r dt + m g e_3 + m \dot{r}_d \\
&\mathbf{b}_{2d} = (\cos \psi_d, \sin \psi_d, 0), \quad \mathbf{b}_{2d} = (b_{2d} \times b_{1d})/\|b_{1d} \times b_{2d}\| \\
\mathbf{R}_d &= (b_{2d} \times b_{3d}, b_{2d}, b_{3d}) \\
f &= \frac{1}{2} (\mathbf{R}_d^T \mathbf{R} - \mathbf{R} \mathbf{R}_d^T)^v, \quad \mathbf{e}_\omega = \omega - \mathbf{R}_d \mathbf{R}_d^T \dot{\omega}_d \\
\mathbf{R}_e &= \mathbf{R}_R \mathbf{e}_R - \mathbf{K}_r \mathbf{e}_r
\end{align*}
$$

(18)

where $(\cdot)^v$ is the usual ‘vee’ map taking a skew-symmetric matrix into the associated 3D vector, and $K_r$, $K_v$, $K_i$, $K_R$, $K_\omega$ are $3 \times 3$ diagonal control gain matrices.

We note that, compared to the original [16], the moment equation (last row of (18)) neglects the compensation for the gyroscopic term in customary in actual implementations of this controller. Note also that, as in the previous near-hovering case, the actual quadrotor inputs $u$ are retrieved by plugging in (17) the thrust/torques $(f, \boldsymbol{\tau})$ computed from (18).

The vector of control gains $k_c$ considered in problem (12–13) for this controller consists of all the 15 gains in (18) (the diagonal entries of $K_r$, $K_v$, $K_i$, $K_R$, $K_\omega$).

D. Sliding Mode Controller (SMC)

The last controller considered in this work is based on sliding mode, which is a control technique known to generally deal well with model uncertain parameters [17]–[19].

The controller is built by following the typical steps of the sliding mode design. We start by considering the vertical $z$ dynamics and define the error $e_z = \dot{r}_d - z$ and the sliding surface $\sigma_z = e_z + \lambda_z e_z$ where $\lambda_z > 0$ is a control gain. The Lyapunov candidate $V_z = 1/2 \sigma_z^2$ has time derivative along the system trajectories

$$
\dot{V}_z = \sigma_z \dot{\sigma}_z = \sigma_z \left( \dot{r}_d - \frac{f}{m} \cos(\phi) \cos(\theta) + g + \lambda_z \dot{e}_z \right).
$$

(19)

We then choose the thrust $f$ as

$$
f = \frac{m}{(\cos(\phi) \cos(\theta))} \left( g + \dot{r}_d + \lambda_1 \tanh(\sigma_z) + \lambda_2 \sigma_z \right)
$$

(20)

where the $\tanh(\cdot)$ provides a smooth approximation of the classical sign(·) used in sliding mode. Following analogous arguments, one can define the torque commands as

$$
\begin{align*}
\tau_x &= k_1 \phi \tanh(\sigma_\phi) + k_2 \sigma_\phi \\
\tau_y &= k_1 \theta \tanh(\sigma_\theta) + k_2 \sigma_\theta \\
\tau_z &= k_1 \psi \tanh(\sigma_\psi) + k_2 \sigma_\psi
\end{align*}
$$

(21)

where $\sigma_\phi = \dot{\phi} + \lambda_\phi \dot{\phi}$, $\sigma_\theta = \dot{\theta} + \lambda_\theta \dot{\theta}$, and $\sigma_\psi = \dot{\psi} + \lambda_\psi \dot{\psi}$ are the sliding surfaces, and $e_\phi = \phi - \dot{\phi}$, $e_\theta = \theta - \dot{\theta}$, and $e_\psi = \psi - \dot{\psi}$. The desired roll/linear angles are defined as in (16) but with

$$
\begin{align*}
\dot{x}_r &= \dot{r}_d + k_1 \dot{\phi}, \quad \dot{y}_r = \dot{r}_d + k_1 \dot{\theta}, \quad \dot{z}_r = \dot{r}_d - r_x, \quad \text{analogously for } \sigma_y.
\end{align*}
$$

(22)

The vector of control gains $k_c$ considered in problem (12–13) consists of all the 18 gains in (20–22), that is, the ‘sliding’ gains $\lambda_\bullet$ and the two gains $(k_{s_1}, k_{s_2})$ for the $x, y, z, \phi, \theta, \psi$ channels.

IV. STATISTICAL ANALYSIS

We now report the results of a statistical analysis for the three controllers of Sect. III when considering the three optimization problems (11–13). For the analysis we generate $N_{\text{traj}} = 25$ initial trajectories $s_d(a, t)$ starting at the origin and coping with the initial/final state constraints and input saturations as in (11–13). These initial trajectories are rest-to-rest motions (from a hovering state to a hovering state) with a final position randomly generated inside a spherical shell of 4 to 6 m centered at the origin, and a final yaw angle randomly generated in the interval $[-\pi/2, \pi/2]$. The trajectories $s_d(a, t)$ are implemented as piecewise Bezier curves as already done in [5], [6], [20] with the goal of achieving a smooth curvature with little snap, as in [21], [22]. The framework is implemented in Python and utilizes the COBYLA [23] nonlinear optimizer from the nlopt toolbox, along with the symbolic toolbox SymEngine to represent the system symbolically. Additionally, the Jitcode [24] framework is employed for in-time compilation of ordinary differential equations. In the given framework, the NH and LEE controllers have optimization timeframes of 2-3 minutes for each trajectory type, while the SMC controller requires 3-4 minutes per trajectory type.

For the sake of exposition, in the following we will let $INIT$ represent an initial (non-optimized) trajectory, $OPT_1$ the corresponding optimized trajectory solution of (11), $OPT_k$ the corresponding optimized trajectory solution of (12) and $OPT_{ak}$ the corresponding optimized trajectory solution of (13). We thus obtain a total of $4N_{\text{traj}}$ trajectories:
of these parameters \( p \) being randomly drawn from (6). The final positions \( \mathbf{r}(t_f) \) of the perturbed trajectories are denoted by small red spheres at \( t = t_f \), forming a point cloud that constitutes the empirical variance ellipsoid centered at \( \mathbf{r}_d(t_f) \).

![Trajectory tracking results for NH, LEE, and SMC controllers](image)

Fig. 2. Trajectory tracking results for NH, LEE, and SMC controllers are shown in the left, center, and right columns respectively, for the INIT, \( OPT_a \), \( OPT_k \), and \( OPT_{ak} \) cases. Each run has \( N_{sim} \) trajectories (green), starting from the origin and ending at the same target location, with the parameters \( p \) being uniformly sampled to lie inside and equal to the shape parameter \( \lambda \) of the ellipsoid (6).

For a first assessment, in Fig. 2 we report the \( N_{sim} \) ‘perturbed runs’ for one INIT trajectory (1st row) and the three corresponding \( OPT_a \) (2nd row), \( OPT_k \) (3rd row) and \( OPT_{ak} \) (4th row) when considering the three controllers of the previous section on every column. The plots reports different target reach \( \mathbf{r}(t_f) \) after each perturbed simulation (in green) forming the empirical variance ellipsoid (red) of the final quadrotor position \( \mathbf{r}(t_f) \) centered at the desired \( \mathbf{r}_d(t_f) \). By referring to Fig. 2 and Fig. 3, one can note how the ellipsoids are reduced in the \( OPT_a \), \( OPT_k \) cases and even further in the \( OPT_{ak} \) cases w.r.t. the INIT cases as expected. The largest radius of the state ellipsoid is decreased on average for the \( OPT_a \) and \( OPT_k \) from (12 cm) to (9 and 7.5 cm) respectively but further decreases in the \( OPT_{ak} \) to (3 cm). This then confirms that the sensitivity cost minimized in (11–13) captures well how variations in the parameters affect deviations in the considered states.

Furthermore, one may note that in the optimized cases \( OPT_a \) and \( OPT_{ak} \), where the shape parameter \( \alpha \) of the reference trajectory is optimized, the optimal trajectory performs an initial maneuver for then arriving at the target location with an almost vertical straight line motion (see Fig. 1 and the attached video for more details). This trend has been empirically verified in almost all the simulation runs and is shared by all the employed controllers. A likely explanation is that a final (almost) vertical motion allows to better minimize the effects of the uncertainties in the considered parameters since the rotational dynamics is not excited during this phase, contrarily to the INIT trajectory where the quadrotor needs instead to perform a non-negligible tilting for reaching \( \mathbf{r}_d(t_f) \) at rest. This insight can then be helpful for designing more robust hover-to-hover motions also in different conditions. This trend can also be deduced from Table I that shows the percentage contribution of each parameter \( (N_p = 4) \) in \( p \) on the sensitivity of the
variables $r(t_f)$ defined by $C_{i,j}$ in (23) (see [25]). This analysis can effectively assess the impact of each parameter variation on the final target reach error at time $t_f$

$$C_{i,j} = \frac{\Pi_i^2(t_f)\delta^2_{p,i}}{\sum_{j=1}^{N_p} \Pi_j^2(t_f)\delta^2_{p,j}} \times 100$$

Table I shows that changes in the parameters $g_x$ and $g_y$ have the most significant impact on the sensitivity of states $x$ and $y$, while state $z$ is predominantly influenced by $k_f$. Consequently, when attempting to reach the final target, a vertical motion is primarily affected only by uncertainties in $k_f$, while the other three parameters affect the states in non-vertical motions. Furthermore, based on Table I, a 15% uncertainty in $k_t$ does not affect the states $r(t_f)$ and can then be disregarded.

Figure 4 reports the actuator speeds (in revolutions per minute) for the $OPT_{ak}$ case. The red line represents the nominal actuator speed, while the green lines represent the actuation speeds in the $N_{sim}$ perturbed runs. The dashed black lines denote the limit bounds of the actuator, and the blue line represents the input bound tubes, which ensures actuation feasibility when $p$ varies in the ellipsoid (6).

Fig. 5 reports the so-called violin plots of the norm of the final target reach (defined as $\|r_d(t_f) - r(t_f)\|$) for the complete statistical campaign considering the three controllers and all the $4N_{traj}$ trajectories in the $N_{sim}$ runs. The interquartile range is indicated by the thick black bar at the center, the median of the samples is shown by the white dot at the center, and the remaining portions of the distribution are indicated by the thin black line. Greater probabilities of the population adopting the specified value are represented by wider areas of the violin plot. In order to obtain a fair comparison of all controllers, we tuned their initial gains so as to have a comparable median in the INIT case, which is around 8 cm in the $INIT$ case. The NH controller average target reach is 7 cm, the LEE controller improvement is about 5 cm, and the SMC controller exhibits the most significant improvement of 4 cm. These results imply that the LEE and SMC controllers are more susceptible to controller gain tuning compared to the NH controller. Moreover, in the $OPT_{ak}$ case, all controllers show significant improvement in both the average target reach and its variance. The median of the NH, LEE, and SMC controllers drops from about 8 cm in the $INIT$ case to around 2 cm for $OPT_{ak}$ case (400% improvement), with a better distribution around the median.

Therefore, from this analysis we can conclude that the SMC controller is not particularly sensitive to the shape of the reference trajectory, but it is quite sensitive to the control gains which, if tuned well, can lead to good performance. The NH and LEE controllers are instead more affected by the trajectory shape in addition to the control gains. Finally, we can note that the NH and LEE controllers perform quite similarly when optimized. Our most likely explanation is that the controllers are evaluated in terms of the target reach $\|r_d(t_f) - r(t_f)\|$ at the final time, and not in terms of how well they are tracking the reference trajectory during motion. The LEE controller should be better at dealing with maneuvers involving, e.g., large tilting or accelerations, but it can have a comparable performance to the NH one when it comes to reach a final hovering state.

V. CONCLUSIONS

In this work we have proposed a novel optimization problem in which the effect of parametric uncertainties in a robot model (quantified by the notion of state sensitivity) can be minimized by acting on the reference trajectory to be tracked and/or on the control gains. A tracking task for a quadrotor UAV has been chosen as case study, by employing three popular tracking controllers widely used in the community. The results of a large statistical analysis have shown the effectiveness of the proposed joint optimization of the reference trajectory and gains in minimizing the effects of uncertainties in selected quadrotor parameters. In addition to identifying the most influential parameters on the system states, we can also gain valuable insights into controller performance and gain tuning robustness concerning parametric uncertainty.
We are now working on an experimental validation of these findings and, on a more methodological side, we are also interested in finding more principled ways to impose bounds on the control gains in problems (12-13) besides simple ‘box constraints’ (e.g., by explicitly considering the stability of the closed-loop via a Lyapunov function).

REFERENCES


